PATTERNS 2011

The Third International Conferences on Pervasive Patterns and Applications

ISBN: 978-1-61208-158-8

September 25-30, 2011
Rome, Italy

PATTERNS 2011 Editors

Fritz Laux, Reutlingen University, Germany

René Reiners, Fraunhofer FIT - Sankt Augustin, Germany

Alfred Zimmermann, Reutlingen University, Germany

PATTERNS 2011

Foreword

The Third International Conferences on Pervasive Patterns and Applications [PATTERNS 2011], held between September 25 and 30, 2011 in Rome, Italy, targeted the application of advanced patterns, at-large. In addition, to support for patterns and pattern processing, special categories of patterns covering ubiquity, software, security, communications, discovery and decision were considered. As a special target, the domain-oriented patterns covered a variety of areas, from investing, dietary, forecast, to forensic and emotions. It is believed that patterns play an important role on cognition, automation, and service computation and orchestration areas. Antipatterns come as a normal output as needed lessons learned.

PATTERNS 2011 was aimed at technical papers presenting research and practical results, industrial small- and large-scale systems, challenging applications, position papers addressing the pros and cons of specific topics, such as those being discussed in the standard fora or in industry consortia, survey papers addressing the key problems and solutions on any of the topics, short papers on work in progress, and panel proposals.

We take here the opportunity to warmly thank all the members of the PATTERNS 2011 Technical Program Committee, as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to PATTERNS 2011. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the PATTERNS 2011 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that PATTERNS 2011 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the area of pervasive patterns and applications.

We are convinced that the participants found the event useful and communications very open. We also hope the attendees enjoyed the charm of Rome, Italy.

PATTERNS 2011 Chairs:

Markus Goldstein, DFKI (German Research Center for Artificial Intelligence GmbH), Germany
Cornelia Graf, CURE - Center for Usability Research & Engineering, Austria
Teemu Kanstren, VTT, Finland
Fritz Laux, Reutlingen University, Germany
Herwig Manaert, University of Antwerp, Belgium
Guenter Neumann, DFKI (Deutsches Forschungszentrum fuer Kuenstliche Intelligenz GmbH), Germany

René Reiners, Fraunhofer FIT - Sankt Augustin, Germany
Zhenzhen Ye, iBasis, Inc., Burlington, USA
Michal Zemlicka, Charles University - Prague, Czech Republic
Alfred Zimmermann, Reutlingen University, Germany

PATTERNS 2011

Committee

PATTERNS Advisory Chairs

Herwig Manaert, University of Antwerp, Belgium
Fritz Laux, Reutlingen University, Germany
Michal Zemlicka, Charles University - Prague, Czech Republic
Alfred Zimmermann, Reutlingen University, Germany

PATTERNS 2011 Research/Industry Chairs

Teemu Kanstren, VTT, Finland
Guenter Neumann, DFKI (Deutsches Forschungszentrum fuer Kuenstliche Intelligenz GmbH), Germany
Markus Goldstein, DFKI (German Research Center for Artificial Intelligence GmbH), Germany
Zhenzhen Ye, iBasis, Inc., Burlington, USA
Cornelia Graf, CURE - Center for Usability Research & Engineering, Austria
René Reiners, Fraunhofer FIT - Sankt Augustin, Germany

PATTERNS 2011 Technical Program Committee

Luis Alberto Álvarez-González, Universidad Austral de Chile - Valdivia, Chile
Abdullah M. Alnajim, Qassim University, Saudi Arabia
Junia Anacleto, Federal University of Sao Carlos, Brazil
Francesca Arcelli, Università degli Studi di Milano-Bicocca, Italy
Senén Barro Ameneiro, University of Santiago de Compostela, Spain
Bernhard Bauer, University of Augsburg, Germany
Noureddine Belkhatir , University of Grenoble, France
Hatem Ben Sta, Université de Tunis - El Manar, Tunisia
Manfred Broy, Technical University Munich, Germany
David W. Bustard, University of Ulster - Coleraine, UK
William Cheng-Chung Chu(朱正忠), Tunghai University, Taiwan
Bernard Coulette, Université de Toulouse 2 - Le Mirail, France
Karl Cox, University of Brighton, UK
Jean-Charles Créput, Universite de Technologie de Belfort Montbéliard (UTBM), France
Mohamed Dahchour, National Institute of Posts and Telecommunications - Rabat, Morocco
Suash Deb, C.V. Raman College of Engineering, India
Vincenzo Deufemia, Università di Salerno - Fisciano, Italy
Kamil Dimililer, Near East University, Cyprus
Mohamed Farouk Abdel Hady, University of Ulm, Germany
Eduardo B. Fernandez, Florida Atlantic University - Boca Raton, USA
Francesco Fontanella, Università degli Studi di Cassino, Italy
Leonardo Garrido, Tecnológico de Monterrey, Mexico
Harald Gjermundrod, University of Nicosia, Cyprus
Markus Goldstein, DFKI (German Research Center for Artificial Intelligence GmbH), Germany
Pascual Gonzalez, University of Castilla-La Mancha – Albacete, Spain

Gustavo González-Sánchez, MEDIAPRO, Spain
Cornelia Graf, CURE - Center for Usability Research & Engineering, Austria
Carmine Gravino, University of Salerno - Fisciano, Italy
Yann-Gaël Guéhéneuc, École Polytechnique de Montréal, Canada
Sven Hartmann, Clausthal University of Technology, Germany
Wladyslaw Homenda, Warsaw University of Technology, Poland
Wei-Chiang Samuelson Hong, Oriental Institute of Technology, Taiwan
Chih-Cheng Hung, Southern Polytechnic State University, USA
Shareeful Islam, University of East London, UK
Maria João Ferreira, Universidade Portucalense, Portugal
Hermann Kaindl, TU-Wien, Austria
Abraham Kandel, University South Florida - Tampa, USA
Teemu Kanstren, VTT, Finland
Alexander Knapp, Universität Augsburg, Germany
Christian Kruschitz, University of Klagenfurt, Austria
Richard Laing, The Robert Gordon University, Aberdeen, UK
Hervé Leblanc, University of Toulouse 3, France
Gyu Myoung Lee, Institut Telecom, Telecom SudParis, France
Alain Léger, Orange - France Telecom R&D & ENSM St. Etienne, France
Haim Levkowitz, University of Massachusetts Lowell, USA
Chendong Li, University of Connecticut, USA
Honghai Liu, University of Portsmouth, UK
Pericles Loucopoulos, Loughborough University, United Kingdom
Herwig Manaert, University of Antwerp, Belgium
Constandinos Mavromoustakis, University of Nicosia, Cyprus
Murali Medidi, Boise State University, USA
Gerrit Meixner, German Research Center for Artificial Intelligence (DFKI) - Kaiserslautern, Germany
Ivan Mistrík, Independent Consultant. Heidelberg, Germany
Paula Morais, Universidade Portucalense Infante D. Henrique - Porto, Portugal
Haris Mouratidis, University of East London, UK
Günter Neumann, DFKI GmbH - Saarbruecken, Germany
Javier Ortega-Garcia, Universidad Autonoma de Madrid, Spain
Ana Paiva, University of Porto, Portugal
Rodrigo Paredes, Universidad de Talca, Chile
João Pascoal Faria, University of Porto, Portugal
Juan Pelaez, U.S. Army Research Laboratory, USA
Christian Percebois, IRIT / Université de Toulouse, France
Leif E. Peterson, Cornell University - Houston, USA
Cuong Pham-Quoc, Ho Chi Minh City University of Technology, Vietnam
José R. Pires Manso, University of Beira Interior, Portugal
Agostino Poggi, University of Parma, Italy
Ramalingam Ponnusamy, Vinayaga Missions University - Tamilnadu, India
Mar Pujol, Universidad de Alicante, Spain
Claudia Raibulet, Università degli Studi di Milano-Bicocca, Italy
Thurasamy Ramayah, Universiti Sains Malaysia - Minden, Malaysia
René Reiners, Fraunhofer FIT - Sankt Augustin, Germany
Joel Rodrigues, Instituto de Telecomunicações, University of Beira Interior, Portugal
José Raúl Romero, University of Córdoba, Spain

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Information Demand Patterns

Capturing Organizational Knowledge about Information Flow

Kurt Sandkuhl
Institute of Computer Science, The University of Rostock, Rostock, Germany
and
School of Engineering, Jönköping University, Sweden
Kurt.Sandkuhl@uni-rostock.de

***Abstract*—The work presented in this paper aims at contributing to reduced information overload in organizations. We propose to capture organizational knowledge about the typical information demand of an organizational role in an information demand pattern, which can serve as basis for analyzing information flow problems and developing organizational and technical improvements. The contributions of this paper are (1) the concept of an information demand pattern, (2) the structure for textual descriptions of such patterns, and (3) experiences from the development and validation process. The approach is based on an industrial case and illustrated by an example pattern. The main achievements are validated research results, i.e. the concept of information demand patterns and actual patterns.**

Keywords: Information demand; organizational knowledge pattern; information logistics; information demand pattern

I. INTRODUCTION

The work presented in this paper aims at contributing to reduced information overload in organizations by capturing reusable organizational knowledge about information demand in so called information demand patterns. Information overload has been a phenomenon observed and discussed in the literature since many decades. One of the pioneers of computer-supported collaborative work, Vannevar Bush, foresaw already in 1945 that it would not be possible to manage all information we collect in our "bewildering store of knowledge" [13]. Simpson and Prusak discussed in 1995 [14] a conceptual model for reducing overload by using quality attributes to information.

Nowadays, information overload is also perceived as a problem in enterprises. Today's enterprise information systems usually support well-defined work flows and routine activities by sophisticated solutions. But for exceptions from daily routine, ad-hoc tasks or seemingly unstructured activities, quickly finding the information needed is a challenge. Some studies show that users spend a lot of time in searching for the right information causing unnecessary delays and costs. An example is [17] showing that 2 out of 3 top or mid-level managers participating in a study among Swedish enterprises perceive an information overload due to "far too much information" (37%) or "too much information" (29%). Since accurate and readily available information is essential in decision making situations, problem solving and knowledge-intensive work, an improved information supply would contribute significantly to saving time and most likely to improving productivity.

The paper aims at contributing to the reduction of information overload by bringing together experiences from organizational knowledge management and pattern-based reuse in information system development. We propose to capture organizational knowledge about the typical information demand of an organizational role in an information demand pattern, which can serve as basis for analyzing information flow problems and developing organizational and technical improvements. The contributions of this paper are (1) the concept of an information demand pattern, (2) the structure for textual descriptions of such patterns including an example, and (3) the development and validation process in an industrial case.

The remaining part of the paper is structured as follows: Section II describes the background for this paper and introduces related work. Section III presents an industrial case motivating the work. Section IV introduces the concept of information demand pattern, including a definition of the term, the structure of pattern descriptions and an example. Section V describes the development process and validation of this pattern. Section VI summarizes the work and discusses future work.

II. BACKGROUND AND RELATED WORK

A. Organizational Knowledge Patterns

Background for the work presented in this paper is research in enterprise knowledge modeling with the objective to make organizational knowledge reusable by applying techniques from enterprise modeling.

In general terms, enterprise modeling is addressing the systematic analysis and modeling of processes, organization structures, products structures, IT-systems or any other perspective relevant for the modeling purpose [18]. Enterprise knowledge modeling combines and extends approaches and techniques from enterprise modeling. The knowledge needed for performing a certain task in an enterprise or for acting in a certain role has to include the context of the individual, which requires including all relevant perspectives in the same model. Thus, an essential characteristic of knowledge models are "mutually reflective views of the different perspectives included in the model"

 ISBN: 978-1-61208-158-8

[8]. Enterprise knowledge modeling aims at capturing reusable knowledge of processes and products in knowledge architectures supporting work execution [19].

In this context, we define the term organizational knowledge pattern as follows:

An organizational knowledge pattern is a formalization of knowledge for a recurring organizational task abstracting from organization-specific aspects, which is of value for an organizational actor and an asset for an organization.

In the context of this definition, the following characteristics of organizational knowledge patterns (OKP) have to be emphasized:

- OKP need to represent organizational knowledge, not individual knowledge, i.e. support the organizational knowledge creation process, the organizational context for use of knowledge by individuals as opposed to supporting knowledge creation of an individual.
- OKP address recurring organizational tasks and at the same time abstracting from a specific organization, i.e. like most other kinds of patterns in computer science is the description of the core elements independent from the actual solution for an organization.
- OKP are expressed in a formalized way, which requires a formal language or at least a structured representation. Thus, OKP are explicit knowledge.
- OKP are an asset of the organization, i.e. are not only a resource as such but capture knowledge about the resource's use. This means they do not only capture how to use the pattern (as for many computer science patterns) but how to use the resource.
- An OKP is of value for an organizational actor in its original form and / or its adaptation for a specific organization.

B. Related Work

For more than a decade, patterns have been popular in computer science and were introduced for numerous areas, like software design, information modeling or business processes. Although there is no generally accepted definition of the term pattern, most publications in the field get some inspiration from Christopher Alexander's definition: "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice" [3]. Whilst Alexander's focus is on the solution, many pattern approaches in computer science focus more on capturing proven practices or an advice on, how to approach certain problems.

The seminal book on patterns was published by the "Gang of Four" [7] and focuses on software design patterns. Many other books followed, basically offering patterns for all phases of the software development process, including analysis patterns [5], data model patterns [8] or software architecture patterns [9]. The pattern idea was adapted in other areas of computer science, like ontology patterns [12] or groupware patterns [10].

Concepts, methods and technologies for identifying, capturing and reusing organizational knowledge have been subject of research in organizational sciences and industrial engineering since more than two decades. Patterns of organizational knowledge, such as task patterns [15] or workflow patterns [11] are a recognized contribution to this area. The term knowledge pattern has been explicitly defined by Clark, Thomson and Porter in the context of knowledge representation [2]. They define "a pattern as a first-order theory whose axioms are not part of the target knowledge-base, but can be incorporated via a renaming of the non-logical symbols" [2, p.6]. The intention is to help construct formal ontologies by explicitly representing recurring patterns of knowledge, so called theory schemata, and by mapping these patterns on domain-specific concepts. Staab et al. [4] investigated the use of so called "semantic patterns" for enabling reuse across languages when engineering machine-processable knowledge. Semantic patterns consist in this approach of one description of the core elements independent from the actual implementation and for each target language a description that allows for translating the core elements into the target language.

III. INDUSTRIAL CASE

The industrial case defining the context for work presented in this paper is taken from automotive industries and focuses on engineering change management in a networked organization with different suppliers. One partner is the business area "seat comfort components" of a first tier automotive supplier with the main product development sites in Scandinavia. The seat comfort products mainly include seat heater, seat ventilation, climate control, lumber support and head restraint. One of the sub-suppliers of this first tier supplier is performing different surface treatment services of the metal components. Surface treatment in this context includes corrosion protection, different technical coatings to achieve certain functionality or decorative coatings in order to reach matt, lustrous or colored surfaces.

As part of the project "infoFLOW", which runs from 2006 - 2011 and includes 7 industrial and academic partners, the engineering change management process between the supplier and sub-supplier was analyzed including the different work steps involved, the roles included on both sides, the information and documents exchanged between both partners, common causes for problems in operations and the needs and goals of improvements. The process is geographically distributed involving engineers and specialists at several locations of the automotive supplier and sub-supplier. A large percentage of the produced components are parts of product families, i.e. various versions of the components exist and have to be maintained and further developed for different customers. In this context, managing different versions of specifications, change requests and production orders including their validity and date of effectiveness is of crucial importance. The results of the process analysis were captured in enterprise models and later used in the development of a method specialized on information demand analysis.

Selected results of analyzing the engineering change process are:

 ISBN: 978-1-61208-158-8

- In both enterprises, many different persons involved in the change management are having similar roles, like on sub-supplier side different key account managers for different projects from the supplier or on supplier side product managers for different customer projects ordering surface specific treatments at the sub-supplier.
- The change management process as such is well-defined and agreed on. However, in daily practice the number of "exceptional" and "emergency" actions is significant and causing many situations where the defined process is not followed.
- Conventional business process improvement approaches were tried, but did not lead to significant improvements, which most likely is based on the large number of exceptions.
- Many change requests are time critical; not implementing them accurately or timely will cause high costs for supplier and sub-supplier, e.g. by coating thousands of metal components wrongly, which will not be accepted by the customer and have to be disposed.
- The roles perceive information overload, and at the same time an information "underload": a lot of task-related information is available and continuously reaching them, but the really relevant information often is "lost in the information flood" or not reaching them.

Based on these results it was decided to use an approach for improving the process based on the information demand of the roles. To find reusable solutions applicable in several enterprises was considered an important goal and given priority in the project.

IV. INFORMATION DEMAND PATTERNS

Starting from a definition of the term information demand (Section IV.A), this section introduces the definition and structure of information demand patterns (Section IV.B) and presents an illustrative example (Section IV.C).

A. Information Demand

The notion of information demand is closely related to work in two areas: information logistics and information retrieval. The main objective of the research field information logistics is improved information provision and information flow [6]. This is based on information demands with respect to the content, the time of delivery, the location, the presentation and the quality of information. The research field information logistics explores, develops, and implements concepts, methods, technologies, and solutions for the above mentioned purpose.

As part of the information logistics research, Lundqvist [1] investigated the nature and characteristics of information demand (ID) in an enterprise context in an empirical study involving 3 organizations and 25 informants. The conclusion from the study is information demand of employees in an organization is to a large extent based on the organizational role, i.e. the information demand of a person is based on the roles and tasks this person has: "*Information demand depends on the role and tasks an entity has within a larger organization. If the role and/or the tasks change, so too will the demand" [1, p. 60]*. This role-centric perspective with task and responsibilities as primary characteristics has been the starting point for developing the approach of information demand patterns.

Information retrieval aims at retrieving relevant information meeting the needs of a user, which are expressed by a query. In this context the aspect of relevance of information is of high importance. Saracevic [16] considers several types of relevance, e.g. algorithmic, topical and cognitive relevance. The underlying concept for algorithmic relevance is the relation between the query features and the search result. Topical relevance is the relation between aboutness of content objects and query. These two relevance concepts are important for retrieving information meeting the demand of a user, but do not contribute to explaining information demand as a concept. Here, we consider cognitive relevance of higher importance, which is the association between perceived information need of the user and information presented to the user based on retrieval results.

B. Definition and Structure of ID-Patterns

The general idea of information demand patterns is similar to most pattern developments in computer science: to capture knowledge about proven solutions in order to facilitate reuse of this knowledge. In this paper, the term information demand pattern will be defined as follows:

An information demand pattern addresses a recurring information demand problem that arises for specific roles and work situations in an enterprise, and presents a conceptual solution to it.

An information demand pattern consists of a number of essential parts used for describing the pattern:

A statement about the *organizational context* where the pattern is useful. This statement usually identifies the application domain or the specific departments or functions in an organization forming the context for pattern definition.

Problems of a role that the pattern addresses. The tasks and responsibilities a certain role has are described in order to identify and discuss the challenges and problems, which this role usually faces in the defined organizational context.

The *conceptual solution* that resolves the problem, which for information demand patterns includes three parts:

- *Information demand* of the role, which is related to the tasks and responsibilities, is described as part of the pattern, i.e. the different parts of the information demand are identified
- *Quality criteria* for the different parts of the information demand include the general importance of the information demand part, the importance of receiving the part completely and with high accuracy, and the importance of timely or real-time information supply
- a *timeline* indicating the points in time when the different information parts should be available at the latest

The *effects* that play in forming a solution. If the needed information part should not be available or arrive too late this might affect the possibility of the role to complete its task

 ISBN: 978-1-61208-158-8

and responsibilities. The effects described in the pattern include

- Potential economic consequences
- Time/efficiency effects, i.e. whether the role will need more time for completing the task or will be less efficient
- Effects on improving or reducing the quality of the work results
- Effects with respect to the motivation of the role responsible
- Learning and experience effects
- Effects from a customer perspective

The above parts of a pattern are described in much detail in the *textual description* of the pattern, which is the main way of representing an information demand pattern (see Section IV.C for an example). Additionally, a pattern can also be represented as a *visual model*, e.g. a kind of enterprise model. This model representation is supposed to support communication with potential users of the pattern, as it illustrates the information demand context, including

- the relation of the role to co-workers and other roles in the organization
- the relation between the different parts of the information demand and IT system in the enterprise, which are potential source of this information
- the relation of tasks and responsibilities to processes in the organization

C. Information Demand Pattern Example

In order to illustrate the pattern approach, the information demand pattern "material specification responsible" was selected. Due to space limitations, only the textual description is presented, i.e. the visual representation is not included. The textual description follows the structure introduced in Section IV.B. The first element is the context where the pattern is useful:

Context:

The context for this pattern is manufacturing industries, in particular automotive industries and automotive suppliers. When developing new products or variants of the existing product families, usually a team of many different engineering disciplines is involved. One role in this team is the material responsible who is supposed to make sure that the applied materials meet the quality requirements of both, the customer of the supplier and the supplier as such. This role requires experience and accurate information about the material characteristics, test results, customer requirements, supplier information etc., which usually originate from different information sources and actors. The pattern describes the information demand typically experienced by the role responsible for contributing to product design processes.

The pattern is supposed to be useful for manufacturing enterprises producing products with high demands to material characteristics developed in distributed teams of engineers.

The next part is the problem addressed by the pattern:

Problem:

The pattern addresses the general problem of unnecessary shortcomings, product design or production problems caused by the materials used. This includes the following problems, which were observed by practitioners in product design projects:

- *Changes in customer requirements affect the required features of the material, but this is difficult to detect in the change description. Thus, the material responsible does not receive this information, although changes in the material or adjustments would be necessary. Long-term effects could be production problems, acceptance failure by the customer or quality problems due to shortcomings in the material.*
- *Test results or policy changes, possibly from other business areas of the company using the same material, indicate that the use of the material should be changed. This information is not reaching the material responsible, as this role is not part of the respective work process or organization unit where the relevant information is produced.*

It follows the information demand, which is based on the task and responsibilities of the role under consideration:

Information Demand

The information demand is based on the tasks and responsibilities of the role. The tasks of the material responsible include

- *Responsibility: the material selected for a certain product version fulfills all desired quality attributes, including customer requirements, internal policy, laws and regulations, sustainability, manufacturability*
- *Establish suitable test methods for the material (for example by selecting the appropriate ones or initiating new developments) and initiate tests continuously validating the desired quality of the material*
- *Specify material characteristics to be used in procurement*

The information demand of the role material responsible consists of:

- *To receive information about all change requests from the customer side that possibly could affect the material of the product under design or manufacturing*
- *To get all information about changes in company-internal policies or in public laws and regulations regarding the use of the material in the company's products*
- *To receive all information about customer complaints or complaints from the production site regarding the material*
- *To get all relevant information from material testing w.r.t. test results of the material in question and regarding modification or new developments regarding the test methods and test processes*
- *To get all information from the supplier side regarding changes in the characteristics of the material*
- *To receive reports and complementary information from certification organization, domain organizations or clusters regarding the material used.*

The quality criteria for the above information demand information uses three levels:

- Decisive: you can't manage without this information
- High: it is very important to have, but in worst case you could complete the task without
- Nice to have: you will manage without this information, but the result will be not as good as with this information

For each pattern, the quality criteria are summarized in a table, which includes the information demand (left column),

the general importance of this information, and the importance to get the information accurately, as soon as possible (here: in close to real-time) and completely. Below is the table for the example pattern:

Information Demand	*General importance*	*Accurate*	*In real-time*	*Complete*
Customer Change Requests	*decisive*	*Decisive*	*high*	*Decisive*
Policy, law, regulation changes	*high*	*High*	*high*	*Decisive*
Complaints from own production	*decisive*	*High*	*decisive*	*High*
Customer complaints	*decisive*	*Decisive*	*high*	*Decisive*
Test results	*high*	*High*	*high*	*High*
Changes on supplier side	*high*	*High*	*high*	*High*
Complementary information	*Nice to have*	*Nice to have*	*Nice to have*	*Nice to have*

Table 1: Quality criteria for the information demand of the role "material specification responsible"

The effects of not receiving the needed information or of receiving it to late is described in a short text and in a table. We will only include an excerpt of the text due to space limitations:

Effects

If the needed information should not be available or arrive too late this will have effects on the work of the material specification responsible:

- *Economic effects: the economic consequences could be: in worst case costs for recall of non-functional parts; increased cost for the component or product, reducing the profit margin for the supplier; increased level of investment in production equipment*
- *Time/efficiency of the task: production or product design will need much more time and will be less efficient. An example is when replacing one failing material, one extra test loop is required for the validation of the complete product*
- *Quality improvement or reduction: the quality of the products is positively or negatively affected by this information. Examples are: late reinforcements of the product, due to failing materials, might result in reduced performance, for instance in mechanical comfort level*
- *[…]*

V. PATTERN DEVELOPMENT AND VALIDATION

The elaboration and validation of the concept of information demand pattern in general and of actual information demand patterns for specific organizational roles was performed in an iterative way consisting of various cycles, including both structure and content. Figure 1 illustrates the overall approach.

As indicated in Section II, the concept of information demand patterns is inspired by other work in computer science. Thus, the initial work on elaboration of the concept "information demand pattern" (marked as "1" in figure 1) focused primarily on the structure of such patterns based on the literature in the field. The validation of the initial structure ("2" in figure 1) consisted of checking this structure for internal consistency and soundness, and using it for capturing a very simple pattern example. The elaboration of the first actual pattern using the initial pattern structure consisted of selecting a simple but sufficiently complex organizational role and developing the content of the pattern for the selected role (step "3"). Afterwards, the initial content of the pattern again was validated by checking internal consistency and completeness, and by using own experiences (step "4"). Steps 1 to 4 form the first iteration, which was performed in a lab environment without involvement from actors outside the research team.

The second iteration started with the next elaboration phase for the pattern structure (step "5"), which was based on experiences from elaborating and validating the first actual pattern (steps 3 and 4). Improvement of the pattern structure and the following step of validation for the improved version (step "6") now also involved additional actors, namely the industrial partners of the infoFLOW project. Later iterations (i.e. steps beyond "8") included elaboration and validation activities involving actors outside the project team and outside academia. The different iterative development steps did not change the structure of patterns significantly, but mainly contributed to a refinement of the level used for describing the quality criteria, the effects and the timeline.

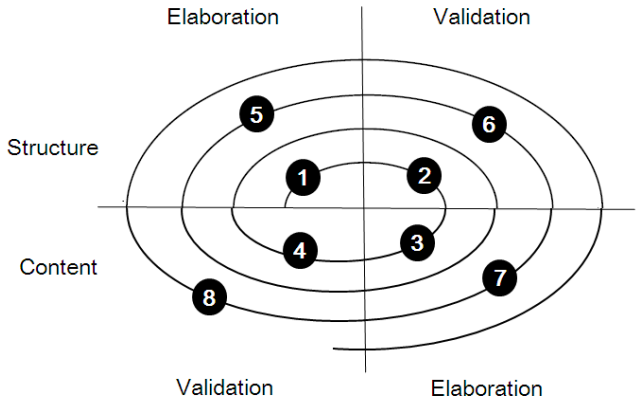


Fig. 1. Iterative process of elaboration and validation of information demand patterns.

The pattern presented in Section IV was developed in the context of the industrial case introduced in Section III by analyzing a number of enterprise knowledge models with focus on reoccurring roles and their relations to processes and infrastructure resources. All enterprise models addressed different parts of collaborative product development in

 ISBN: 978-1-61208-158-8

networked enterprises. The focus of the analysis was to identify roles reoccurring in the different models and indicating a specific information demand. The selected role "material specification responsible" appeared in several models developed. The information demand of these roles was derived from the textual descriptions accompanying the enterprise models and the scenario descriptions developed in the modeling process.

The pattern was validated in two steps: The first version was presented, discussed and refined during an infoFLOW project meeting. This included a walkthrough the visual model and a in-detail discussion of the textual description. The revised version was presented to an industrial expert in the field who proposed changes and improvements, primarily regarding the effects to be expected and the required information quality. This refined version was again discussed in a project meeting.

VI. SUMMARY AND FUTURE WORK

The paper presented an approach for capturing organizational knowledge about the information demand for an organizational role in information demand patterns. Starting from task and responsibilities of a role, these patterns identify the information needed for the role under consideration, the required level of accuracy, completeness and timeliness, a typical timeline, and the economic, quality, efficiency and customer effects of receiving the information too late of in insufficient quality.

In their current development state, information demand patterns are supposed to contribute to explaining and investigating operational problems in an organization caused by information lack of a role:

- Patterns help to analyze and identify information demand problems. If an organization detects information problems in a certain part of the organization or intends to identify improvement potentials, a pattern can be used to compare the typical or desirable solution described in the pattern with the current situation in the organization
- Patterns identify and specify abstractions that are above the level of single solutions in specific enterprises and provide a common vocabulary and understanding for information demand problems.
- Patterns are a means of documenting solutions to information flow problems.
- Patterns help to manage information supply complexity by dividing larger information flow problems in smaller "modules", which can be expressed in patterns.

Future work will focus on four main aspects

- Increasing the number of patterns, applying them in organizations and evaluating the benefits of the information demand pattern approach
- Extending the patterns toward solutions for information supply, i.e. how can the information demand described in a pattern be supported by IT-solutions or organizational measures
- Investigate and define the process of adaptation of an information demand pattern to another context of use
- Development of a method for pattern development and refinement, including technical support for pattern management, like a pattern repository.

ACKNOWLEDGMENT

The research presented was financed by the Swedish Knowledge Foundation (KK-Stiftelsen), grant 2009/0257, project "Demand Patterns for Information Logistics (infoFLOW-2)".

REFERENCES

[1] Lundqvist, M. (2007) *Information Demand and Use: Improving Information Flow within Small-scale Business Contexts.* Licentiate Thesis, Dept of Computer and Information Science, Linköping University, Linköping, Sweden, ISSN 0280-7971.

[2] Clark, P.; Thompson, J., and B. Porter (2000). Knowledge Patterns. In A. G. Cohn, F. Giunchiglia, and B. Selman, editors, KR2000: Principles of Knowledge Representation and Reasoning, San Francisco, 2000. Morgan Kaufman.

[3] Alexander, C. et al. (1977) A Pattern Language. Oxford, 1977.

[4] Staab, S., Erdmann, M., and Maedche, A. (2001) Engineering Ontologies using Semantic Patterns. In D. O'Leary and A. Preece (eds.) Proceedings of the IJCAI-01 Workshop on E-business & The Intelligent Web, Seattle, 2001.

[5] Fowler, M. (1997) Analysis Patterns. Addison Wesley, 1997.

[6] Sandkuhl, K. (2008) Information Logistics in Networked Organizations: Selected Concepts and Applications. Enterprise Information Systems, 9th ICEIS 2008. LNBIP, Springer.

[7] Gamma, Helm, Johnson, and Vlissides (1995) Design Patterns. Addison-Wesley, 1995.

[8] Hay (1995) Data Model Patterns. Dorset House, 1995.

[9] Buschmann et al. (2000) Pattern-oriented Software Architecture. Wiley.

[10] Schümmer, T. and S. Lukosch (2007) *Patterns for Computer-Mediated Interaction*, Wiley & Sons, ISBN: 978-0-470-02561-1, 2007.

[11] Van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., and A.P. Barros (2003) Workflow Patterns. *Distributed and Parallel Databases*, 14, 5–51, Kluwer.

[12] Blomqvist E. and K. Sandkuhl (2005) *Patterns in Ontology Engineering – Classification of Ontology Patterns*. Proc. 7th International Conference on Enterprise Information Systems, Miami, USA, May 2005.

[13] Bush, V. (1945) "As we may think" *The Atlantic Monthly*, July.

[14] Simpson, C.W. and L. Prusak (1995) "Troubles with Information Overload – Moving from Quantity to Quality in Information Provision", *International Journal of Information Management*, Vol. 15 No. 6, pp. 413-425.

[15] Sandkuhl, K. (2008) Capturing Product Development Knowledge with Task Patterns: Approach and Economic Effects. Proceedings IMS 2008, Szczecin, Poland, IFAC paper online.

[16] Saracevic, T. (1996) Relevance Reconsidered ´96. In Ingwersen, P.; Pors, N. O. (eds.): Information Science: Integration in Perspective. Royal School of Library and Information Science, Copenhagen, Denmark, pp. 201-218, 1996.

[17] Öhgren, A. and K. Sandkuhl (2008) Information Overload in Industrial Enterprises - Results of an Empirical Investigation. Proceedings ECIME 2008, London, UK.

[18] Vernadat, F.B. (1996) *Enterprise Modelling and Integration*. Chapman & Hall, 1996.

[19] Lillehagen F. and D. Karlsen (1999). Visual Extended Enterprise Engineering embedding Knowledge Management, Systems Engineering and Work Execution. *IFIP International Enterprise Modelling Conf.*, Norway.

 ISBN: 978-1-61208-158-8

A Pattern Language for Architecture Assessments of Service-oriented Enterprise Systems

Alfred Zimmermann
Faculty of Informatics
Software Architecture Research Group
Reutlingen University, Germany
alfred.zimmermann@reutlingen-university.de

Fritz Laux
Faculty of Informatics
Data Management Research Group
Reutlingen University, Germany
fritz.laux@reutlingen-university.de

René Reiners
Fraunhofer FIT
Schloss Birlinghoven
Sankt Augustin, Germany
rene.reiners@fit.fraunhofer.de

***Abstract* – The SOA Innovation Lab – an innovation and research network of industry leaders in Germany and Europe - investigates the practical use of service-oriented enterprise systems. Current state of practice approaches for assessing maturity of service-oriented enterprise software architectures were intuitively developed, having sparse metamodel or pattern foundation and being rarely validated. This is a real problem for practical architecture assessments in repeatable cyclic evaluations of base architectures of service-oriented systems, which are based on recurring patterns for analysis of continuously growing services over the time. In our research we have developed an original pattern language for supporting architecture assessments and optimization of enterprise systems, leveraging and extending base frameworks like the Capability Maturity Model Integration (CMMI) and The Open Group Architecture Framework (TOGAF), and considering additionally similar related work models of architecture maturity. We have deduced our original architecture quality assessment and optimization pattern language from our special designed architecture maturity framework. We have applied our architecture assessment pattern language in consecutive cyclic assessment workshops with global vendors of service-oriented platforms.**

Keywords – service-oriented architecture; architecture assessment; pattern language; metamodel; maturity framework.

I. INTRODUCTION

Innovation oriented companies have introduced service-oriented architectures (SOA) to assist in closing the business - IT gap. Our approach investigates the SOA ability of heterogeneous enterprise systems as in [1] and integrates elements from convergent architecture methods, technologies and related software patterns, as in [2], [3], and [6] with evaluation methods for service-oriented enterprise systems as in [4].

The hypothesis of our research is as follows:

1) The CMMI [5] is well known as a suitable framework to assess software processes; nevertheless the metamodel of CMMI can be extended to enable quality assessments of service-oriented software architectures.

2) The idea of software patterns could be applied consistently in architecture assessments for both capability assessments and improvements of service-oriented architectures for enterprise systems.

Software architecture assessment patterns are based on the seminal work of software patterns originated from the work of [7]. A pattern records solution decisions taken by many builders in many places over many years in order to resolve a particular problem. Patterns are human readable structures of text and graphics showing a standardized and repeatable way to derive a solution from a specified problem in a specific context. Patterns describe usually sophisticated and hidden solutions for given design problems. But we can also use patterns like testing patterns for the assessment of software architectures and other software artifacts. We call a collection of patterns, which are organized in a directed acyclic graph structure, a *pattern language*. Elements of a pattern language are navigable sequences of patterns.

Our original and validated pattern language approach for assessment patterns for quality assessments of enterprise software architectures relies on related work (Section II) and on a specific maturity framework (see Section III) for assessing architecture capabilities and maturity levels of service-oriented enterprise systems. We derived an associated architecture assessment pattern language from our previous work on an architecture maturity framework and our basic architecture quality pattern catalog, which was also previously developed. The new introduced architecture quality assessment language extends and sequences the basic architecture assessment patterns to 43 integrated patterns (Section IV). These patterns are used to identify quality indicators for different architectural aspects and specific structures of service-oriented software systems. Finally, we sketch main evaluation results (Section V) and draw conclusions and future directions (Section VI).

II. RELATED WORK

The Open Group Architecture Framework (TOGAF) [8] as the current standard for enterprise architecture provides the basic blueprint and structure for our enterprise software architecture domains of service-oriented enterprise systems like *Architecture Strategy and Management, Business Architecture, Information Architecture, Application Architecture, Technology Architecture, Service & Operation Architecture,* and *Architecture Realization*.

SOA is the computing paradigm that utilizes services as fundamental flexible and interoperable building blocks for both structuring the business and for developing applications. SOA promotes a business-oriented architecture style as

 ISBN: 978-1-61208-158-8

promoted in [9] and [3]), based on best of breed technology of context agnostic business services that are delivered by applications in a business focused granularity. To provide dynamic composition of services within a worldwide environment SOA uses a set of XML-based standards. A main innovation introduced by SOA is that business processes are not only modeled, but services are executed from orchestrated services, too.

To transform CMMI into a specific framework for architecture assessments of service-oriented enterprise systems we have combined CMMI with current SOA frameworks and maturity models. We use TOGAF [8] and ideas related to the business and information architecture from [10] as a basic structure for enterprise architecture spanning all relevant levels of service-oriented enterprise systems.

The Architecture Capability Maturity Model (ACMM) [11] framework, which is included in TOGAF, was originally developed by the US Department of Commerce. The goal of ACMM assessments is to enhance enterprise architectures by identifying quantitative weak areas and to show an improvement path for the identified gaps of the assessed architecture. The ACMM framework consists of six maturity levels and nine specific architecture elements ranked for each maturity level - deviant from CMMI.

The SOA Maturity Model of Inaganti and Aravamudan [12] considers the following multidimensional aspects of a SOA: scope of SOA adoption, SOA maturity level to express architecture capabilities, SOA expansion stages, SOA return on investment, and SOA cost effectiveness and feasibility. The scope of SOA adoption in an enterprise is differentiated by the following levels: intra department or ad hoc adoption, inter departmental adoption on business unit level, cross business unit adoption, and the enterprise level, including the SOA adoption within the entire supply chain. The SOA maturity levels are related to CMMI, but used differently , using five ascending levels to express enhanced architectural capabilities: level 1 for initial services, level 2 for architected services, level 3 for business services, level 4 for measured business services, and level 5 for optimized business services.

The SOA Maturity Model from Sonic [13] distinguishes five maturity levels of a SOA, and associates them in analogy to a simplified metamodel of CMMI with key goals and key practices. Key goals and key practices are the reference points in the SOA maturity assessment.

The SOA Maturity Model of ORACLE [14] characterizes in a loose correlation with CMMI five different maturity levels: opportunistic, systematic, enterprise, measured, industrialized and associates them with strategic goals and tactical plans for implementing SOA. Additionally the following capabilities of a SOA are referenced with each maturity level: Infrastructure, Architecture, Information & Analytics, Operations, Project Execution, Finance & Portfolios, People & Organization, and Governance.

III. ASSESSMENT FRAMEWORK

Our idea and contribution is to extend existing service-oriented architecture (SOA) maturity frameworks to accord with a sound metamodel approach. Our metamodel for architecture evaluation enlarges the standardized CMMI, which is originally used to assess the quality of software processes and not the quality of software architectures.

The aim of our SOAMMI – SOA Maturity Model Integration - framework [15] is to provide a holistic framework to assess architectures of service-oriented enterprise systems. We have analyzed and systematically integrated evaluation criteria, maturity domains, architecture capabilities, and level rankings from state of the art SOA maturity and evaluation models as described in [11], [13], and [14]. In addition we have adapted architecture assessment elements from [4] and [15], and extended singular architecture patterns from our previous work [16] to our new assessment pattern language (Section IV).

The SOAMMI architecture maturity framework introduces original architecture areas and organizes them within extended architecture domains, which are mainly based on TOGAF. Our intention was to leave most structural parts e.g. *Maturity Levels, Capability Levels, Specific Goals and Practices, Generic Goals and Practices* - of the original CMMI metamodel as untouched concepts. We extend these concepts of the metamodel by reclusively connected architecture patterns, as navigable architecture quality patterns of a pattern language, and enlarge these by other architecture specific structures and contents. The metamodel of SOAMMI in Figure 1 has similarities with the CMMI metamodel. We leaved the understanding of Maturity Levels and Capability Levels the same like in CMMI. Additionally we added the following concepts: Architecture Domain, Architecture Area, Architecture Pattern, and replaced all the contents of related Specific Goals, Specific Practices, and the Generic Practices, to fit for our architecture evaluation purpose. We used multiplicity indicators for class relations to add a basic metamodel semantic. Not indicated multiplicities corresponds to the default 1 cardinality or a 1..1 multiplicity.

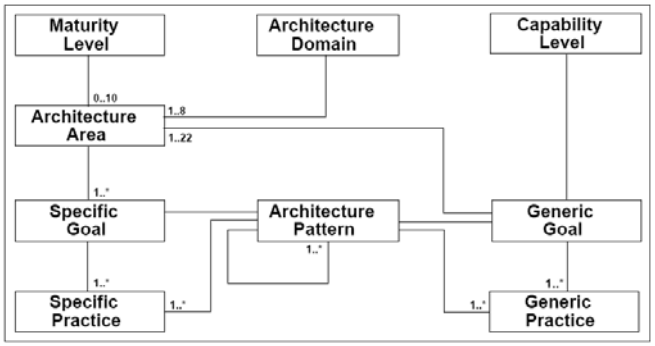


Figure 1. SOAMMI Metamodel – Main Concepts.

In terms of requirements from customer oriented domain-models and reference use scenarios, our model has introduced in [15] the following maturity levels, which define architecture assessment criteria for service-oriented enterprise systems and help to measure the architecture maturity: The semantic of these maturity levels as in [15]

 ISBN: 978-1-61208-158-8

was adapted from [5] to conform with the architecture assessment scope for service-oriented enterprise systems:

1. *Maturity Level: Initial Architecture,*
2. *Maturity Level: Managed Architecture,*
3. *Maturity Level: Defined Architecture,*
4. *Maturity Level: Quantitatively Managed Architecture,*
5. *Maturity Level: Optimizing Architecture.*

We have derived the architecture domains mainly from TOGAF [8], where they are used as specific architecture subtypes and corresponding phases of the TOGAF-ADM (Architecture Development Method). Architecture areas cover assessable architecture artifacts and are correspondent, but very different, parts of process areas from CMMI. To fit our architecture assessment scope, we have defined 22 original architecture areas of the SOAMMI framework [15], linked them to our architecture maturity levels and ordered them in line with our specific enterprise and software architecture domains. Each of the delimited architecture area is accurately described in a catalog including *name* of architecture area, *short identification* of architecture area and a *detailed description*.

SOAMMI supports both the staged and continuous representations. The same staging rules as in CMMI apply to SOAMMI and should therefore enable the flexible adoption of both model representations: *continuous* for assessing single architecture areas and *staged* for assessing the whole architecture maturity. The assessment of capability levels could be applied to iterate specific architecture areas or to assess or improve a focused innovation aspect, involving one ore more architecture areas. To verify and support persistent institutionalizations of architecture areas we introduce architecture related generic goals and practices. All architecture areas are affected by the same generic goals and associated generic practices. In the following, two example architecture areas together with their goals and practices are presented.

A. Examples of Architecture Areas

1. Business Products & Services

Purpose: Structure, design, model, and represent business products and associated business services, which are necessary to support modeled products.

Maturity Level: 3

Specific Goals (SG) and Specific Practices (SP):

SG 1: Model Business Products as Origin of Business Processes

- SP 1.1 Structure business products within product lines
- SP 1.2 Design business products by defining product structures and product rules
- SP 1.3 Model and represent business products

SG 2: Model Business Services associated with Business Products

- SP 2.1 Structure business services according to product types
- SP 2.2 Design business services by defining service structures and service levels
- SP 2.3 Model and represent business services

The second example extends contents from the first example and provides a base for our pattern language scenario:

2. Business Processes & Rules

Purpose: Structure, design, model, and represent business value chains and business processes to support business capabilities.

Maturity Level: 2

Specific Goals (SG) and Specific Practices (SP):

SG 1: Model Business Value Chains as Root of Business Capabilities and Business Processes

- SP 1.1 Identify business value for business operations
- SP 1.2 Structure value chains
- SP 1.3 Optimize business considering customer channels and supplier networks

SG 2: Model and Optimize Business Processes

- SP 2.1 Identify business activities for business processes: system activities, user interaction activities, manual activities
- SP 2.2 Structure business processes for business roles and organizational units
- SP 2.3 Define business workflows and business process rules
- SP 2.4 Model and represent business processes

SG 3: Model and Represent Business Control Information

- SP 3.1 Identify and represent control information for product monitoring
- SP 3.2 Identify and represent control information for process monitoring.

IV. ASSESSMENT PATTERN LANGUAGE

Our pattern language for architecture assessments of service-oriented enterprise systems provides a procedural method framework for architecture assessment processes and questionnaire design. This method framework of our new introduced pattern language was inspired from [7], and derived from the structures of the metamodel of SOAMMI as well as from our seminal pattern catalog from previous research [16]. We note that our architecture patterns are basically assessment process patterns for enterprise architecture management and are therefore not fine granular classical design patterns.

We have linked each specific and each generic goal within our assessment framework to a distinctive pattern of our pattern language. We organize and represent our architecture assessment patterns according to the following structures: *Architecture Domains*, *Architecture Areas*, *Problem Descriptions* - associated with *Specific Goals*, *Solution Elements* - are connected to *Specific Practices*, *Related Patterns* - are connections to next applicable patterns of the pattern language.

Linking solution elements to specific practices of the SOAMMI framework enables concrete solutions for architecture assessments and improvements of service-oriented enterprise systems. This assessment and improvement knowledge is both verification and design knowledge, which is a procedural knowledge based on standards, best practices, and assessment experience for

 ISBN: 978-1-61208-158-8

architecture assessments of service-oriented enterprise systems. It is therefore both concrete and specific for setting the status of service-oriented enterprise architectures, and helps to establish an improvement path for change. Patterns of our language show what to assess. Our patterns aim to represent verification and improvement knowledge to support cooperative assessments synchronizing people in cyclic architecture assessments.

Associated with our architecture assessment pattern language we have set up an assessment process to show how to assess architecture capabilities. This process is based on a questionnaire for architecture assessment workshops providing concrete questions as in [4], answer types, and helping to direct and standardize the related assessment process. Additionally, we have included process methods for workshops, result evaluations, improvement path information for technology vendors and for application organizations, as well as change support and innovation monitoring instruments.

Based on the two examples of architecture areas from Section III - Business Products & Services Architecture and Business Process & Rules, we sketch a pattern language scenario (as a typical small example):

1. Model and represent Business Products
2. Model Business Services for Business Products
3. Model Business Value Chains as Root of Business Processes
4. Model and Optimize Business Processes
5. Model and Represent Business Control Information.

We are representing the core causalities of our architecture assessment and improvement patterns in the reduced canonical form, which we have adapted from [2] and [6]. Our pattern form denominates consciously the problem and the solution part as basic elements of our patterns. This canonical form is extendable in further work by additional parts like contexts, forces, examples, explanations, and linked patterns. The following examples show a concrete extract of 5 related patterns, which derives from the sketched architecture areas from Section III:

1. *Pattern Example: Business Product*
   Problem: How can we structure, design, model, and represent each business product as an origin for modeling business processes?
   Solution:
   - Structure business products for product lines
   - Design business products by defining product structures and product rules
   - Model and represent business products

   Related Patterns: Business Services, Value Chain, Business Process, Business Control Information
2. *Pattern Example: Business Service*
   Problem: How can we structure, model, and represent each business service needed to support business products?
   Solution:
   - Structure business services for product types
   - Design business services by defining service structures and service levels
   - Model and represent business services

   Related Patterns: Value Chain, Business Process, Business Control Information
3. *Pattern Example: Value Chain*
   Problem: How can we structure, optimize and represent business value chains as roots for business process modeling?
   Solution:
   - Identify business value for business operations
   - Structure value chains
   - Optimize business considering customer channels and supplier networks

   Related Patterns: Business Process
4. *Pattern Example: Business Process*
   Problem: How can we structure, model and optimize business processes, related workflows, and business process rules?
   Solution:
   - Identify business activities for business processes: system activities, user interaction activities, manual activities
   - Structure business processes for business roles and organizational units
   - Define business workflows and business process rules
   - Model and represent business processes

   Related Patterns: Business Control Information
5. *Pattern Example: Business Control Information*
   Problem: How can we model and represent business monitoring and control information?
   Solution:
   - Identify and represent control information for product monitoring
   - Identify and represent control information for process monitoring

   Related Patterns: None

This scenario of pattern interactions is specified by the graph of architecture patterns (Figure 2), indicating the navigation direction (to the next applicable patterns).

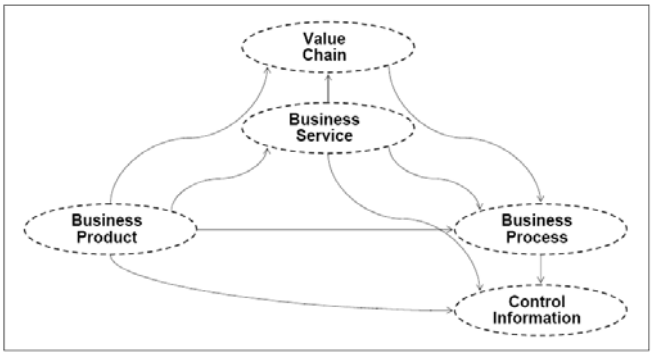


Figure 2. Architecture Pattern Language Scenario.

 ISBN: 978-1-61208-158-8

In Figure 2, we indicated selective navigation paths to the next applicable patterns. Value Chain is for instance only a constructive artifact and not a real observable concept: Therefore the Value Chain Pattern has no associated Business Control Information Pattern. We have identified and distinguish a set of 43 patterns as parts of a new researched and introduced pattern language in the context of 7 Architecture Domains and 22 Architecture Areas. Even though our architecture quality patterns accords to the Specific Goals, the Specific Practices and the Generic Goals from the SOAMMI framework, they extend these structures by navigable patterns as part of an architecture assessment language. Only this pattern structure enables architecture quality assessors to navigate easily in two directions to support the diagnostics and optimization process,, and to provide a clear link to questionnaire and the related answer and result concepts. The full collection of patterns of the architecture assessment pattern language was derived from the SOAMMI framework in (Section III):

1. Architecture Domain: Architecture Strategy and Management

Architecture Area: ***EAM E***nterprise ***A***rchitecture ***M***anagement

1. Pattern: Architecture Strategy

Architecture Area: ***GOV A***rchitecture ***Go***vernance

2. Pattern: Architecture Management
3. Pattern: Architecture Governance

Architecture Area: ***OPM O***rganizational ***P***erformance ***M***onitoring

4. Pattern: Performance Baselines for Architecture

Architecture Area: ***QAM Q***uantitative ***A***rchitecture ***M***anagement

5. Pattern: Manage Architecture Quantitatively
6. Pattern: Manage Architecture Agility

Architecture Area: ***AID A***rchitecture ***I***nnovation and ***D***eployment

7. Pattern: Architecture Innovation Management

Architecture Area: ***CAR C***ausal ***A***nalysis and ***R***esolution

8. Pattern: Causes of Architecture Defects
9. Pattern: Resolution of Architecture Defects

Architecture Area: ***ARM A***rchitecture ***R***equirements ***M***anagement

10. Pattern: Architecture requirements Management

Architecture Area: ***ARD A***rchitecture ***R***equirements ***D***evelopment

11. Pattern: Customer Requirements
12. Pattern: Architecture Requirements
13. Pattern: Validate Architecture Requirements

2. Architecture Domain: Business Architecture

Architecture Area: ***BDC B***usiness ***D***omains & ***C***apabilities

14. Pattern: Business Domain
15. Pattern: Domain Granularity & Coupling

Architecture Area: ***BCS B***usiness ***C***apabilities & ***S***ervices

16. Pattern: Business Capabilities
17. Pattern: Business Services
18. Pattern: Service Agility

Architecture Area: ***BPS B***usiness ***P***roducts & ***S***ervices

19. Pattern: Business Product
20. Pattern: Business Service

Architecture Area: ***BPR B***usiness ***P***rocesses & ***R***ules

21. Pattern: Value Chain
22. Pattern: Business Process
23. Pattern: Business Control Information

3. Architecture Domain: Information Architecture

Architecture Area: ***DEC D***ata ***E***ntities & ***C***omponents

24. Pattern: Entity Service

Architecture Area: ***BOB B***usiness ***O***bjects

25. Pattern: Business Object
26. Pattern: Business-IT-Alignment

4. Architecture Domain: Application Architecture

Architecture Area: **SDO S**ystem ***Do***mains

27. Pattern: System Domain Mapping

Architecture Area: ***SSC S***ystem ***S***ervices & ***C***apabilities

28. Pattern: Application Service Design
29. Pattern: Application Vendor Services

5. Architecture Domain: Technology Architecture

Architecture Area: ***PFS P***lat***f***orm ***S***ervices

30. Pattern: Platform Service Design
31. Pattern: Platform Vendor Services

Architecture Area: ***TSC T***echnology ***S***ervices & ***C***apabilities

32. Pattern: Technology Service Design
33. Pattern: Technology Vendor Services

6. Architecture Domain: Service & Operation Architecture

Architecture Area: ***SDT S***ervice ***D***esign & ***T***ransition

34. Pattern: Support Service Design
35. Pattern: Service Management

7. Architecture Domain: Architecture Realization

Architecture Area: ***ASC A***rchitecture ***S***tandards & ***C***ompliance

36. Pattern: Architecture Standards Management
37. Pattern: Architecture Standards Definition

Architecture Area: ***ACO A***rchitecture ***Co***ntracts

38. Pattern: Architecture Contracts.

Architecture Area: ***AIN A***rchitecture ***In***stitutionalization

39. Pattern: Base Architecture Practices
40. Pattern: Managed Architecture
41. Pattern: Defined Architecture
42. Pattern: Quantitatively Managed Architecture
43. Pattern: Optimizing Architecture.

V. EVALUATION AND FINDINGS

The practical benefits of our pattern language were demonstrated by the successful use as guideline for the questionnaire design in four major capability assessments of service-oriented vendor technology architectures. Architecture assessments need to address key challenges for companies during the built-up and management of service-oriented architectures.

SOAMMI seems to be complex in practice. Therefore a pragmatic simplification of the SOAMMI framework was

particularly required in counting assessment results. Additionally we have considered for our assessments specific user requirements from companies using and providing service-oriented enterprise systems.

Following these ideas, the basic structure of our questionnaire [15] was taken from the SOAMMI architecture areas with one or more questions per Specific Goal. User requirements have been consolidated and mapped against specific goals. Wherever no user requirements could be mapped, Specific Practices have been used to generate questions on the level of specific goals. Through this procedure each Specific Goal could be related to at least one concrete question.

The assessment process takes about 3 months in total to complete for each software technology provider. The first step is a pre-workshop (2-3 hours) to make sure that the architecture provider can identify the appropriate experts for the assessment workshop itself. Then the actual assessment workshop (4- 6 hours) is held a few weeks later, so that the provider has enough time to identify the experts that should participate and prepare answers. Finally, a series of follow up workshops for specific questions (3-4 hours each) are arranged with the system technology provider.

VI. CONCLUSION AND FUTURE WORK

We have introduced an original pattern language for assessing capabilities and architecture maturity of service-oriented enterprise systems. In this paper we have motivated the necessity to extend existing SOA maturity models to accord to a clear metamodel approach due to the verified CMMI model. Based on the related work to CMMI, which is an assessment and improvement model for software processes but not for architectures, we have developed suitable models for assessments of service-oriented enterprise systems. Our specific architecture assessment approach of the SOAMMI framework was founded on current architecture standards like TOGAF and architecture assessment criteria from related work approaches. Additionally a dashboard was developed to support practical assessment processes, which were aligned both with the process for CMMI and with empirical questionnaire and interview methods. The presented SOAMMI framework was validated in consecutive assessment workshops with four global vendors of service-oriented platforms and has provided transparent results for subsequent changes of service oriented product architectures and related processes. Our empirical validation and optimization of the presented maturity framework is an ongoing process, which has to be synchronized with future cyclic evaluations of SOA platforms and their growing number of services. Extended validations of customers of service oriented technologies are planned for the next phase of our framework research and development. Future work additionally has to consider conceptual work on both static and dynamic architecture complexity, and in connecting architecture assessment procedures with prognostic processes on architecture maturity with simulations of enterprise and software architectures. Additional improvement ideas include patterns for visualization of architecture artifacts and architecture control information to be operable on an architecture management cockpit. We are working at extending our pattern language to a full canonical form in order to support full standardized cyclic architecture assessments for service-oriented products and solutions.

REFERENCES

[1] H. Buckow, H.-J. Groß, G. Piller, K. Prott, J. Willkomm, and A. Zimmermann, "*Method for Service-Oriented EAM with Standard Platforms in Heterogeneous IT Landscapes*", Proc. 2nd European Workshop on Patterns for Enterprise Architecture Management (PEAM2010), Paderborn, Germany, GI-Edition - Lecture Notes in Informatics (LNI), P-160, 2010, pp. 219-230.

[2] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, "*Design Patterns*", Addison Wesley, 1994.

[3] T. Erl, "*SOA Design Patterns*", Prentice Hall. 2009.

[4] P. Bianco, R. Kotermanski, and O. Merson, "*Evaluating a Service-Oriented Architecture*", SEI-2007-TR-015, Carnegie Mellon University, Software Engineering Institute, 2007.

[5] CMMI-DEV-1.3 2010 "*CMMI for Development, Version 1.3*", Carnegie Mellon University, Software Engineering Institute, SEI-2010-TR-033, 2010.

[6] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, „*Pattern-oriented Software Architecture*", Wiley, 1996.

[7] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson., I. Fiksdahl-King, and S. Angel, „*A Pattern Language*", Oxford University Press, 1977.

[8] TOGAF "*The Open Group Architecture Framework*" Version-9, The Open Group, 2009.

[9] D. Krafzig, K. Banke, and D. Slama, „*Enterprise SOA*", Prentice Hall, 2005.

[10] *Essential Architecture Project*, http://www.enterprise-architecture.org, last access: June, 19th, 2011

[11] ACMM, "*Architecture Capability Maturity Model*", in TOGAF Version 9, The Open Group Architecture Framework, The Open Group, 2009, pp. 685-688.

[12] S. Inaganti and S. Aravamudan, "*SOA Maturity Model*", BP Trends, April 2007, 2007, pp. 1-23.

[13] Sonic: "*A new Service-oriented Architecture (SOA) Maturity Model*", http://soa.omg.org/Uploaded%20Docs/SOA/SOA_Maturity.pdf, last access: June, 19th, 2011.

[14] Oracle: "*SOA Maturity Model*", http://www.scribd.com/doc/2890015/oraclesoamaturitymodel cheatsheet, last access: June, 19th, 2011.

[15] H. Buckow, H.-J. Groß, G. Piller, K. Prott, J. Willkomm, and A. Zimmermann, "*Analyzing the SOA-ability of Standard Software Packages with a dedicated Architecture Maturity Framework*", EMISA 2010: October 7– 8, 2010 - Karlsruhe, Germany, GI-Edition - Lecture Notes in Informatics (LNI), P-172, 2010, pp. 131-143.

[16] A. Zimmermann, E. Ammann, and F. Laux, „*Pattern Catalog for Capability Assessments and Maturity Evaluation of Service-oriented Enterprise Architectures*", PATTERNS 2010 - The Second International Conferences on Pervasive Patterns and Applications, November 21-26, 2010 - Lisbon, Portugal, IARIA Proceedings of the PATTERNS 2010 Conference, 2010, pp. 13-19.

 ISBN: 978-1-61208-158-8

A Novel Live 3D Objects Reconstruction Scheme

Hui Zheng, Jie Yuan, Sidan Du
School of Electronic Science and Engineering, Nanjing University
Nanjing, China
Email: hzheng.nju@gmail.com, yuanjie@nju.edu.cn, coff128@nju.edu.cn

***Abstract*— In this paper, we describe a novel method to create the complete 3D model of the object on uncalibrated images. First, we match the points both detected by multi-scale Harris corner detection algorithm and line detection technique. Second, we perform a projected reconstruction based on factorization using Singular Value Decomposition (SVD). After that, we are able to upgrade from projective to Euclidean structure and then eliminate the ambiguity in Euclidean reconstruction. Finally, we use 3D registration algorithm based on common points to build the whole 3D model of the object. Sufficient experiments proved the validity and efficiency of the method.**

Keywords-3D reconstruction; ambiguity in Euclidean Reconstruction; 3D Registration; line detection

I. INTRODUCTION

Reconstruction of the object in 3D from images taken by uncalibrated cameras has long been a topic of research in computer vision. Factorization has been a common and reliable method for 3D reconstruction and motion recovery. Sturm and Triggs [1] proposed a projective reconstruction algorithm based on factorization, they recovered projective depths by estimating a set of fundamental matrixes. Ponce [2] upgraded the projective reconstruction to Euclidean reconstruction on the assumption that the cameras are zero-skew. Mei Han and Takeo Kanade recovered the shape and motion from image sequence which taken by uncalibrated camera using factorization method in [3]. R Szeliski mentioned the ambiguity problem in the process of simultaneously recovering structure and motion with uncalibrated cameras in [4], this ambiguity would cause errors in 3D registration.

Displaying after meshing the 3D models is one of the important applications for 3D reconstruction. Direct feature points detection algorithm [9] [10] would have edge and corner information lost during 3D reconstruction, which would cause the unsatisfactory display. To solve this problem, the method describe in [5] using the feature points which are detected both by Harris and Sift [9] for reconstruction, but it would not reduce the edge information loss.

In this paper, the line detection algorithm and the feature points detection and matching algorithm are proposed in Section II. In Section III, we introduce the projective reconstruction and Euclidean reconstruction in Subsection A and Subsection B, and then propose an algorithm to eliminate ambiguity in Euclidean reconstruction after identifying its cause in Subsection C. In order to create the complete 3D model, we introduce the detail of 3D registration technique in Section IV. Experimental processes and results are introduced in Section V; we use four cameras for synchronized taking $n(n > 2)$ pictures for the moving object, and then reconstruct each of the four parts of the object from the images taken by each camera, and finally create the complete 3D model. In Section VI, we give a conclusion of this paper. There are three contributions in this paper:

- Propose a 3D reconstruction method using line detection technique that ensures the correctness and completeness of corners and edges on the 3D model.
- Describe the cause of ambiguity in Euclidean reconstruction and propose a method to eliminate ambiguity.
- Increase the quantity of common points in 3D using guided matching algorithm that enhance accuracy of 3D registration.

II. FEATURE POINTS DETECTION AND MATCHING

In this paper, we use the multi-scale Harris corner detection algorithm [8] to detect feature points on images. This algorithm is the combination of Harris operator and scale space theory. We first detect the Harris feature points in different scales, and then using LOG operator to select the appropriate scale and obtain the location of the feature points.

The feature points detected by traditional detection algorithm [5] [9] [10] cannot cover the edges and corners of the object, which will cause the loss of edge and corner information when reconstructing the 3D model. So we detect lines on images using line detection algorithm [6] based on Hough transform, and then match the points on the lines for reconstruction.

In this paper, we use the wide baseline matching algorithm [11] to match the points which are detected on the images taken by adjacent cameras, we call it "match between cameras"; we use the algorithm based on guided matching to match the points which are detected on the images taken by one camera, we call it "match in camera".

There are two steps in "matching between cameras". First, we execute initial matching. Calculate the correlation coefficient of feature points of the two images respectively. When the correlation coefficient is larger than a given threshold value, both the feature points are considered as the candidate of matching points. Search support strength from the neighborhood to accumulate matching strength. We merely consider the maximum support of each neighborhood as the initial matching points. Next, we use RANSAC method to calculate the fundamental matrix F and homography matrix H, and remove the mismatches at the same time.

There are also two steps in "matching in camera". We execute initial matching at first. Next, we using guided

 ISBN: 978-1-61208-158-8

matching algorithm [12] to find the points on other images that correspond to the feature points obtained from matching points between cameras and detected lines. Guided matching is a technique which can redirected match the designated feature points by using epipolar geometry and homography constraint.

The outline of feature points detection and matching is as follows:

Suppose $I_j^i (i = 1 \dots n)$ is the image taken by jth$(1 \leq j \leq 4)$ camera at time i.

a) P_j^i =detect feature points on I_j^i

b) match between cameras:
$\{PO_{12}^n\ PO_{21}^1\} = \text{match}\{P_1^n\ P_2^1\}$;
$\{PO_{23}^n\ PO_{32}^1\} = \text{match}\{P_2^n\ P_3^1\}$;
$\{PO_{34}^n\ PO_{43}^1\} = \text{match}\{P_3^n\ P_4^1\}$;
$\{PO_{41}^n\ PO_{14}^1\} = \text{match}\{P_4^n\ P_1^1\}$;

c) line detection:
$m = \left\lfloor \frac{n}{2} \right\rfloor$;
$PL_1^m = line\ detect\{I_1^m\}$;
$PL_2^m = \text{line detect}\{I_2^m\}$;
$PL_3^m = \text{line detect}\{I_3^m\}$;
$PL_4^m = \text{line detect}\{I_4^m\}$;
PL are the points spaced selected on detected lines

d) match in camera:
Camera1:
$\{PI_1^1\ PI_1^2 \dots PI_1^n\} = \text{initial match}\ \{P_1^1\ P_1^2 \dots P_1^n\}$;
$\{PIL_1^1\ PIL_1^2 \dots PIL_1^n\} =$
guided match PL_1^m on I_1^m to $I_1^i (i \neq m)$;
$\{PIO_{12}^1\ PIO_{12}^2 \dots PIO_{12}^n\} =$
guided match PO_{12}^n on I_1^n to $I_1^i (i \neq n)$;
$\{\text{PIO}_{14}^1\ \text{PIO}_{14}^2 \dots \text{PIO}_{14}^n\} =$
guided match PO_{14}^1 on I_1^1 to $I_1^i (i \neq 2)$;
$PM_1^i = PI_1^i \cup PIL_1^i \cup PIO_{12}^i (i = 1 \dots n)$;
$\{PM_1^1\ PM_1^2 \dots PM_1^n\}$ are final matching points of Camera 1. Camera 2, camera 3 and camera 4 do the similar way with Camera 1.

III. 3D RECONSTRUCTION

In order to create the complete 3D model, we have to reconstruct each of the four parts of the object. In Section III, we will describe the reconstruction algorithm.

A. Projective Reconstruction

Suppose there are m object points $X_j (j = 1 \dots m)$ and n perspective matrixes $P_i (i = 1 \dots n)$, x_{ij} is the projection of X_j that projected by P_i.

$$W = \begin{bmatrix} \lambda_{11}x_{11} & \lambda_{12}x_{12} & \cdots & \lambda_{1n}x_{1n} \\ \lambda_{21}x_{21} & \lambda_{22}x_{22} & \cdots & \lambda_{2n}x_{2n} \\ \vdots & & \vdots & \\ \lambda_{m1}x_{m1} & \lambda_{m2}x_{m2} & \cdots & \lambda_{mn}x_{mn} \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_m \end{bmatrix} \begin{bmatrix} X_1 X_2 \cdots X_n \end{bmatrix} = PX \quad (1)$$

In (1), W is scaled measurement matrix and λ_{ij} is a non-zero factor called projective depth. The goal of projective reconstruction is to estimate the projective depth of x_{ij}. We use the algorithm based on SVD described in [3] [7] to estimate λ_{ij}:

a) Set $\lambda_{ij} = 1$. Compute the current scaled measurement matrix W, by (1)

b) Perform SVD decomposition on W : $UDV = \text{SVD}(W)$.

c) Set $P' = U_4, X' = D_4 * V_4^T$ where U_4, D_4, V_4^T are the first four lines of U, D, V.

d) Set $W' = P' * X'$, update λ_{ij} and W:

e) $\lambda_{ij} = \lambda_{ij} * \frac{W_{ij}'^T * W_{ij}}{W_{ij}^T * W_{ij}}$; $W_i = \lambda_{ij} * x_{ij}$

f) Go to step 3 until D(5,5) is small enough.

B. Euclidean Reconstruction

The factorization of (1) recovers the motion P and shape X up to a 4×4 linear projective transformation H:

$$W = P' \times X' = (PH) \times (H^{-1}X) \quad (2)$$

The goal of Euclidean reconstruction is to calculate the matrix H that upgrades the P and X in projective space to P' and X' in Euclidean space. The P_i' in Euclidean space can be written as:

$$P_i' = \alpha_i K * [R_i | T_i] \quad (3)$$

where α_i is a none-zero real number; R_i is a orthogonal matrix, which shows the rotation of the camera; T_i is a vector, which shows the position of the camera. Because of the orthogonality of R_i, we rewrite H as $H = [A|B]$, where A is 4×3 and B is 4×1. We have:

$$P_i A A^T P_i^T = \alpha_i{}^2 K R_i R_i^T K^T = \alpha_i{}^2 K K^T \quad (4)$$

In this paper we only discuss the case that camera pixels are square, and we shift principal point to the original, so the intrinsic parameters matrix K can be denoted by:

$$K = \begin{bmatrix} a_x & 0 & 0 \\ 0 & a_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5)$$

where $a_x = a_y$. Substituting (5) into (4):

$$M_i = P_i A A^T P_i{}^T = P_i Q P_i{}^T = \begin{bmatrix} \alpha_i^2 a_x^2 & 0 & 0 \\ 0 & \alpha_i^2 a_y^2 & 0 \\ 0 & 0 & \alpha_i^2 \end{bmatrix} \quad (6)$$

where

$$Q = AA^T. \quad (7)$$

 ISBN: 978-1-61208-158-8

From (6) we can obtain the equation:

$$M_i(0,0) = M_i(1,1)$$
$$M_i(1,2) = M_i(1,3) = M_i(2,3) = 0 \quad (8)$$

We can set up 4 equations from each frame, and given $\alpha_1 = 1$, we have $4n + 1$ equations to solve Q which have 10 unknown elements. Then we get the matrix A from Q by SVD decomposition.

To solve B we denote $P'$by $P' = [F|T] = P[A|B]$ and substitute $P = (P_x\ P_y\ P_z)^T$, $T = (T_x\ T_y\ T_z)^T$ into it :

$$T_x = P_x * B;\ T_y = P_y * B;\ T_z = P_z * B \quad (9)$$

Put the origin of the world coordinate system at the center of gravity of the scaled object points to enforce:

$$\frac{T_{xi}}{T_{zi}} = \frac{\sum_{j=1}^{m} \lambda_{ij} * x_{ij}}{\sum_{j=1}^{m} \lambda_{ij}} \qquad \frac{T_{yi}}{T_{zi}} = \frac{\sum_{j=1}^{m} \lambda_{ij} * y_{ij}}{\sum_{j=1}^{m} \lambda_{ij}} \quad (10)$$

From (9) (10) we set up 2n liner equations to solve B.

After A and B have been computed, we can obtain motion matrix P' and shape matrix X' in Euclidean space.

C. Eliminate Ambiguity in Euclidean Reconstruction

There exists an ambiguity in Euclidean Reconstruction we described in Section III Subsection B. The 4×3 matrix A that we obtained from Q using SVD decomposition is non-unique. Suppose A is a solution of (7), we have:

$$Q_{ij} = A_{i1} * A_{j1} + A_{i2} * A_{j2} + A_{i3} * A_{j3} \quad (11)$$

From (11) we know A still a solution if we reverse any columns of it:

$$Q_{ij} = (\mp A_{i1}) * (\mp A_{j1}) +$$
$$(\pm A_{i2}) * (\pm A_{j2}) + (\pm A_{i3}) * (\pm A_{j3}) \quad (12)$$

So, there are at most 8 solutions for equation $Q = AA^T$. Substitute A into equation $P' = [F|T] = P \times [A|B]$:

$$F_{ij} = P_{i1} * A_{1i} + P_{i2} * A_{2i} + P_{i3} * A_{3i} \quad (13)$$

Moreover, P' and X' satisfy:

$$P' \times X' = W \quad (14)$$

From (13) and (14) we know that we will obtain different motion matrix P' and shape matrix X' corresponding to different solutions of (7). Suppose A_1 and A_2 are different solutions of (7), and the jth$(j < 4)$ column of A_1 has opposite signs to the jth column of A_2, then each members of jth column of P_1' and P_2' computed from A_1 and A_2 is opposite numbers to each other. Also according to (14) we know that the jth row of X_1' and X_2' has opposite signs corresponding to P_1' and P_2'. When we perform Euclidean reconstruction using the method mentioned in Section III Subsection B, we will obtain one solution randomly from all solutions which satisfy (11) and (14).

Suppose P_a' and X_a' are the solution sets which satisfy (7) (13) and (14).We can divide X_a' into two groups X_l' and X_r'. Every solution in one group can transform to every other one in this group through rotation and translation, but the solutions in different groups cannot transform to each other like that. We can image this situation as every solution $X_l^{i'}$ $(X_l^{i'} \in X_l')$ is in the left-handed coordinate system, and $X_r^{i'}$ $(X_r^{i'} \in X_r')$ is in the right-handed coordinate system.

We must make sure all shapes that are reconstructed from the images which taken by different cameras are in the same type of coordinate system when we generate the complete 3D model of the object using 3D registration algorithm. Figure 1 shows the wrong result that doing 3D registration with two shapes that in different groups.

Suppose I_j^i is the image taken by jth camera at time i, $P_j^{i\prime}$ is the motion matrix of jth camera at time i, X_j' is the shape matrix which reconstruct from I_j^i. If we have the images $\{I_j^{t_1} \dots I_j^{t_1}\}$ and $\{I_j^{t_2} \dots I_j^{t_2}\}(j = 1, 2, 3, 4)$ which are taken by each camera at different moment t_1 and t_2, the rotational direction of the object from t_1 to t_2 computed from each motion matrix $P_j^{i\prime}(i = t_1, t_2; j = 1 \dots 4)$ must be the same(when static cameras capture videos of moving object, the motions that obtained from motion matrixes of cameras have opposite directions with the motion of object). For example, if the object rotate clockwise from t_1 to t_2, the rotational directions that we computed from $P_j^{t_1\prime}$ and $P_j^{t_2\prime}$ must be counter-clockwise. But if we have the wrong situation like Figure 1, each rotational direction obtained from each motion matrixes must be the opposite, so we can eliminate ambiguity according to this property. From (6) we compute intrinsic parameters matrix K as:

$$a_x = a_y = \sqrt{\frac{M_i(0,0) + M_i(1,1)}{2a_i}} \quad (15)$$

We eliminate the ambiguity of Euclidean reconstruction after normalize $P_j^{i\prime}$:

a) Normalize $P_j^{i'}(i = t1, t2)$:
  if $\left(P_j^{i'}(1,1) < 0\right)$
  $P_j^{i'}(:,1) = -P_j^{i'}(:,1) \quad X_j'(1,:) = -X_j'(1,:)$
  if $\left(P_j^{i'}(2,2) < 0\right)$
  $P_j^{i'}(:,2) = -P_j^{i'}(:,2) \quad X_j'(2,:) = -X_j'(2,:)$

b) Compute K_j and $\{R_j^{t1}, R_j^{t2}\}$ from (15) and (3).

c) Compute rotational direction RD_j from $\{R_j^{t1}, R_j^{t2}\}$.

d) Compare $RD_j(1 < j \leq 4)$ with RD_1:

e) if $\left(RD_j \text{ has the opposite direction to } RD_1\right)$
  $P_j^{i'}(:,3) = -P_j^{i'}(:,3) \quad X_j'(3,:) = -X_j'(3,:)$

 ISBN: 978-1-61208-158-8

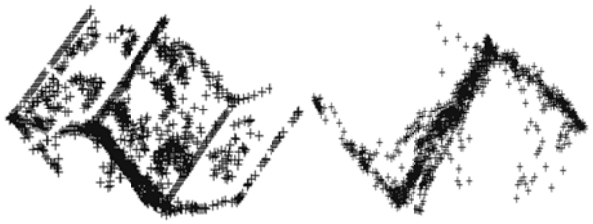


Figure 1. Wrong solution because perform 3D registration with two shapes in different groups.

IV. 3D REGISTRATION

Suppose $\{PO_{12}^n\ PO_{21}^1\}$ $\{PO_{23}^n\ PO_{32}^1\}$ $\{PO_{34}^n\ PO_{43}^1\}$ and $\{PO_{41}^n\ PO_{14}^1\}$ are matchings between cameras which we obtained in Section II. And we can find $\{X_{12}\ X_{21}\}$ $\{X_{23}\ X_{32}\}$ $\{X_{34}\ X_{43}\}$ $\{X_{41}\ X_{14}\}$, which are 3D points corresponding to them.

We take X_{12} from Camera 1 and X_{21} from Camera 2 as example to describe 3D registration. The transformation between X_{12} and X_{21} is:

$$X_{12} = s \times R \times X_{21} + t \quad (16)$$

where s is non-zero number called scale factor, R is rotation matrix, t is translation vector. We compute R as:

$$R = U * diag\,(1,\ 1,\ 1,\ det\,(UV')) * V' \quad (17)$$

We compute U, D, V in (17) from

$$UDV = svd\left(\sum_{i=1}^{n}(X_{12_i} - X_{12A})(X_{21_i} - X_{21A})'\right) \quad (18)$$

where $X_{12A}\ X_{21A}$ are the center of gravity of $X_{12}\ X_{21}$:

$$X_{12A} = \frac{1}{n}\sum_{i=1}^{n} X_{12_i};\ X_{21A} = \frac{1}{n}\sum_{i=1}^{n} X_{21_i} \quad (19)$$

We obtain s from

$$s = \frac{C'}{\sum_{i=1}^{n} ||X_{21_i} - X_{21A}||^2} \quad (20)$$

where C' in (20) compute as (21).

$$C' = D(1,1) + D(2,2) + det(UV') * D(3,3). \quad (21)$$

We obtain t by substituting s, R into (22).

$$t = \frac{1}{n}\sum_{i=1}^{n} X_{12_i} - S * R * \frac{1}{n} * \sum_{i=1}^{n} X_{21_i} \quad (22)$$

Finally we accomplish 3D registration according to (23).

$$Xb_2' = s \times R \times X_2' + t \quad (23)$$

V. EXPERIMENTAL ANALYSIS AND RESULTS

In this Section, we will show the processes of our experiments and analyze the results.

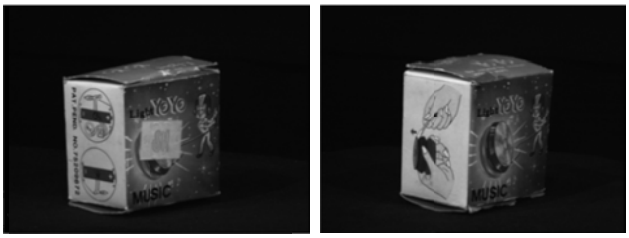


Figure 2. Original images

We use four static uncalibrated cameras to take photos simultaneously of the moving object. Each camera takes three photos on the object with small-scale movement. Figure 2 shows two of the images taken by different cameras. We implement our method in MATLAB, and then display the model with texture in OpenGL.

Reprojective error is an important criterion which represents the reconstruction accuracy. Table 1 shows the main reprojective error of middle image of each part which are reconstructed from images taken by each camera after 3D registration. We can see the reconstruction error of our method is small and acceptable. Because we translate the other parts to part 1's coordinate system in our experiment, the mean reprojective error in part 1 is a little lower than other part.

In order to enhance the accuracy of 3D registration, we using guided matching technique in step 'match in camera' to obtain more common points for 3D registration. Table 2 compares the mean registration error and quantity of common points obtained in 3D registration with guided matching technique with those obtained without guided matching technique. We transform the 3D coordinate into a normalized one before registration, in which each component (x, y or z) of 3D points falls into $[-1, 1]$. From Table 2, we can see our method obtain more common points than the method without guided matching technique, and the mean registration error computed by our method is much smaller than the other one.

TABLE 1. MEAN REPROJECTIVE ERROR AFTER 3D REGISTRATION

	Part1	Part 2	Part 3	Part 4
Mean error (pixel)	0.154	0.362	0.406	0.572
Max error (pixel)	1.118	1.220	1.966	2.34

TABLE 2. MEAN REGISTRATION ERROR AND QUANTITY OF COMMON POINTS IN REGISTRATION

	Obtain common points with guided matching		Obtain common points without guided matching	
	Quantity	Mean error	Quantity	Mean error
Part 1 with Part 2	103	0.0129	11	0.0259
Part 1 with Part 3	122	0.0109	23	0.0207
Part 2 with Part 3 and 4	197	0.0198	24	0.0551

 ISBN: 978-1-61208-158-8

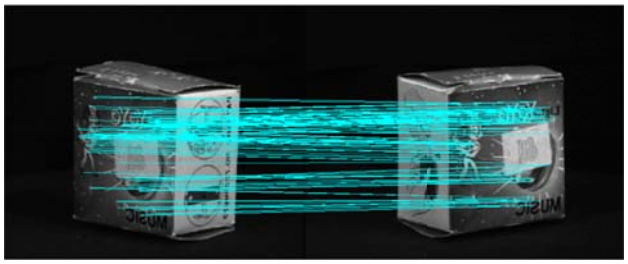


Figure 3. Match between groups

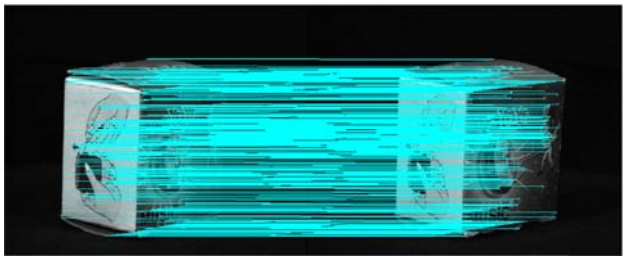


Figure 4. Match in groups

Figure 3 shows the result of 'matching between cameras'. The points in there are matched accurately each other and there are enough matching points for 3D registration. Figure 4 shows one result of "matching in camera". There are three parts matching points in this figure: the points detected by multi-scale Harris corner detection algorithm are the basic matchings; the points on lines and corners prevent the edge and corner information loss of the 3D model; the points corresponding to the matchings in "matching between cameras" step increase the quantity of common points in 3D, which lead to the increase the accuracy of 3D registration.

Figure 5 shows the detected lines on image with white color. Figure 6 shows the mesh of the 3D points generated by performing 3D triangulation algorithm. From that we can see the edges and corners of the box are accurate and complete.

Figure 7 shows the result of 3D reconstruction without line detection technique. There is a lot of edge and corner information loss of the model and the display effect is unsatisfactory.

Figure 8 shows the 3D model reconstructed by our method. Apparently our method prevents the edge and corner information loss effectively and shows a satisfactory display.

Figure 5. Image with detected lines

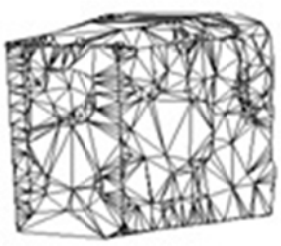


Figure 6. 3D mesh

Figure 7. Reconstruct without line detection technique

Figure 8. Reconstruction result of our method

VI. CONCLUSION

Several conclusions can be drawn from the experiment results. Firstly, line detection algorithm using in our method insures the accuracy and completeness of the edges and corners on reconstructed model. Secondly, the algorithm we proposed in Section III eliminates the ambiguity in Euclidean reconstruction effectively. Thirdly, since we increase the quantity of common points in 3D using guided matching technique, we obtained high accuracy results of 3D reconstruction and 3D registration. The display effect of the model reconstructed from our method is satisfactorily.

VII. ACKNOWLEDGEMENT

This paper is supported by the Fundamental Research Funds for the Central Universities, Number: 1107021051, and National Natural Science Funds of Jiangsu Province of China, Number: BK2010386.

REFERENCES

[1] Peter Strum and Bill Triggs, "A factorization based algorithm for multi-image projective structure and motion," In European conf. Computer Vision — ECCV '96, Cambridge, U.K., 1996, pp. 709-720.

[2] Jean Ponce, "On Computing Metric Upgrades of Projective Reconstructions under the Rectangular Pixel Assumption," Proc. Workshop. 3D Structure from Multiple Images of Large-Scale Environments (SMILE'00), pp. 52-67, 2002.

[3] Mei Han and Takeo Kanade, "Creating 3D Models with Uncalibrated Cameras," Proc. IEEE Workshop. Application of Computer Vision (WACV2000), pp.178-185, 2000.

[4] Richard Szeliski. "Computer Vision: Algorithms and Applications," Chapter 7, 2010, pp. 343-374

[5] Keju Peng, Xin Chen, Dongxiang Zhou, and Yunhui Liu, "3D reconstruction based on SIFT and Harris feature points," Proc. IEEE Conf. Robotics and Biomimetics (ROBIO'09), pp. 960-964, December 2009

 ISBN: 978-1-61208-158-8

[6] Thuy Tuong Nguyen, Xuan Dai Pham, and Jae Wook Jeon, "An improvement of the Standard Hough Transform to detect line segments," Proc. IEEE Conf. Industrial Technology (ICIT'08), Chengdu, China, pp. 1-6, 2008.

[7] Toshio Ueshiba and Fumiaki Tomita, "A Factorization Method for Projective and Euclidean Reconstruction from Multiple Perspective Views via Iterative Depth Estimation," Proc. European Conf. Computer Vision, pp. 296-310, 1998.

[8] Krystian, Mikolajczyk and Cordelia Schmid, "Scale & Affine Invariant Interest Point Detectors," International Journal of Computer Vision, 2004, pp. 63-68.

[9] David G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision. 2004, pp. 91-110.

[10] Zhiyong Ye, Yijian Pei, and Jihong Shi, "An Adaptive Algorithm for Harris Corner Detection," Proc. IEEE Conf. Computational Intelligence and Software Engineering (CiSE'09), pp. 1-4, December 2009.

[11] Jiangjian Xiao and Mubarak Shah, "Two-frame wide baseline matching," Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on. 2003, pp. 603-609.

[12] Benjamin Ochoa and Serge Belongie, "Covariance propagation for guided matching," Proc. Workshop. Statistical Methods in Multi-Image and Video Processing (SMVP'06). On CD-ROM, 2006.

 ISBN: 978-1-61208-158-8

Application of Feature Point Detection and Matching in 3D Objects Reconstruction

Jie Yuan, Ting Feng, Yu Zhou, Yao Yu
School of Electronic Science and Engineering, Nanjing University, Nanjing 210093, China
Email: yuanjie@nju.edu.cn, fengnju1988@gmail.com, nackzhou@nju.edu.cn, allanyu@nju.edu.cn

***Abstract*— In this paper, we present a novel wide baseline based approach to detect and match feature points in image series. We found the wide baseline correspondence problem with large scale, rotation, illumination and affine transformations is still not tackled very well. We proposed a new matching method which based on multi-scale Harris algorithm and two-way guided matching to achieve large number of accurate point correspondences between un-calibrated image sequences of the same scene for wide baseline. We apply our method in the experiments of 3D object reconstruction with satisfied results. It shows that the guided matching method can be used for severe scene variations and provide evidence of improved performance with respect to the SIFT distance and Harris matchers. It is also useful to the matching in short baseline, and the results of this method are better than that of the traditional method.**

Keywords-feature points; image matching; 3D object reconstruction

I. INTRODUCTION

Wide Baseline Matching (WBM) is one of the most important issues that have been extensively studied in the field of computer vision, as well as the foundation of many computer vision theory and applications [1][2][3] such as object identification, camera calibration, 3D reconstruction, and motion analysis. Meanwhile, WBM is a bottleneck in the field of computer vision research. Therefore, research on WBM is of significant importance. WBM primarily divides into two parts: feature point detection and matching.

The primary methods of feature point detection are Harris feature point extraction algorithm and Scale Invariant Feature Transform (SIFT [14]) feature point extraction algorithm. Both of them have their own advantages and disadvantages. To get better corner detection results, Keju [6] combined the two algorithms during 3D reconstruction on demand. But this method has limited the range of application, which means that it is not applicable if we merely to get more and more accurate feature points. Schmid [5] reported that corner extraction algorithm, being invulnerable to camera pose and sunlight, performs the best currently. However, as for vision systems with large scale changing, this method can hardly guarantee invariability of the feature points. In this case, the paper provides a novel multi-scale corner detection approach which combines scale space theory and Harris feature point detection algorithm.

To match feature points, generally, the relative methods are used to achieve the correspondence of two images' point sets. Considering noise interference, light conditions, and other factors which may result in a great number of mismatches, however, removing mismatches is essential. One of the direct ways to remove mismatches is to find an affine transformation which is applicable to all the feature points in the image, and then use it to pre-estimate the position of these feature points located in the other image [4]. Nevertheless, it is not applicable to complicated scenes. To solve this problem, Ferrari et al. [3] proposed to estimate local affine transformation matrix for every pair of corresponded feature points, using least mean square method. Later it was suggested in another approach that this affine transformation matrix should be compared to the predetermined threshold to gather the most similar points to the affine transformation matrix. Although this method has been proven to be effective in confirming mismatches, it pays a high cost of computational complexity. Currently, a comparatively better method in the field of removing mismatches is to use epipolar geometry restriction proposed by Zhang [9]. This method can produce excellent results on the condition that matching points are less in quantity and parallax is small. However, there are two issues remained to be solved. One is that the quantity of matching points is relatively small. The other is the restriction of disparity. Increasing the disparity means enlarging the match searching window, while enlarged match searching window will probably introduce mismatches.

In response to the above problems, we propose a novel approach combining epipolar geometry, homograph constraint, mismatch detection and guided matching which, to some extent, greatly make up the deficiency in these two areas mentioned above. At first, we use relative method to conduct initial match of the image feature points set. Secondly, we use RANSAC (Random Sample Consensus) method to estimate fundamental matrix and homography matrix and remove mismatches in correspondence. Then, we remove mismatches again according to euclidean distance. Finally, we use optimized fundamental and homography matrix guiding the matching to get more and more accurate matching points.

In 3D objects reconstruction, the quality of WBM will affect the result of reconstruction directly. The purpose of this paper is to get a better algorithm in WBM, and apply it to 3D reconstruction. Then the accuracy of 3D objects reconstruction will be enhanced, and fewer cameras or video cameras will be used in experiment.

 ISBN: 978-1-61208-158-8

II. FEATURE POINT DETECTION

An effective feature point detection algorithm is introduced in this section. We introduce the scale space theory at first.

A. Scale Space Theory

Scale space theory is carried out by scaling the original image to obtain multi-scale sequence of scale space, and extracting the main contour based on the sequence as a feature vector, to achieve edge detection, corner detection and feature extraction on different resolutions. As an important concept in scale space theory written by Lindbergh [12][13] that scale space is describing original image at different levels, each layer has a scale parameter which may be discrete and also can be continuous. All scales of space should have the following properties:

- All the signals should be defined in the same domain.
- With the growth of scale parameters, the output image is increasingly blurred.
- Details contained at the coarse level of a signal are less than that at the fine level. If the local maximum is a measure of smoothness, as the scale blurred, extreme non-increasing, this property is known as the "scale space causality".
- All that is generated by a convolution operator.

Scale space kernel is defined as:

$$L(x, y, \sigma) = K(x, y, \sigma) \times I(x, y) \quad (1)$$

In (1), $I(x, y)$ is the original image, σ is the scale parameter. For all the images I, if the extremes of the image $L(x, y, \sigma)$ obtained after its convolution with transform kernel K is less than the extremes of original image, then we call K the scale space kernel. Generally we only use the Gaussian kernel as the scale convolution. Because in Gauss scale space, fine-scale information on the parameter value with the increase in scale was inhibited in the scale of the change from coarse to fine process, no new structure. However, since Gauss kernel is linear, translation invariant, rotation invariant, has subset features and many other properties, it can be proved that Gauss kernel is the only transform kernel to achieve scale space transformation [8]. Feature points and edges of the same type at different scales have a causal relationship, which means that when scale changes, new feature points may arise while old ones may be displaced or disappear. The ambiguity brought by the causal relationship is inherent and inevitable which should never be expected to be eliminated but it can be decreased.

B. Multi-Scale Harris Feature Point Detection Algorithm

Multi-scale Harris feature point detection algorithm was introduced and the experimental results of this method was given in this section.

1) Harris Operator of Scale Space

Harris operator R can be represented as:

$$R = det(C) - ktr^2(C) \quad (2)$$

In (2), $C(x) = \begin{bmatrix} I_u^2(x) & I_{uv}(x) \\ I_{uv}(x) & I_v^2(x) \end{bmatrix}$, k is empirical value, which usually between 0.04~0.06. To obtain the presentation of Harris operator, I_u, I_v can respectively be represented as：

$$I_u(x, s\sigma_n) = I_u(x) * G_u(x, s\sigma_n) \quad (3)$$

$$I_v(x, s\sigma_n) = I_v(x) * G_v(x, s\sigma_n) \quad (4).$$

Then the $C(x)$ function of Harris algorithm will become

$$\hat{C}(x, \sigma_I, \sigma_D) = \sigma_D^2 G(\sigma_I) * \begin{bmatrix} I_u^2(x, \sigma_D) & I_{uv}(x, \sigma_D) \\ I_{uv}(x, \sigma_D) & I_v^2(x, \sigma_D) \end{bmatrix} \quad (5)$$

In (5), $\sigma_I = \sigma_n$ is the selected scale parameter to calculate feature points; $\sigma_D = s\sigma_n$ is the differential scale; $G(\sigma_I)$ is Gaussian function. Through judging to detect the feature corner under σ_n scale level

$$R = det(\hat{C}) - ktr^2(\hat{C}) > T \quad (6).$$

2) Multi-Scale Harris Feature Point Detection Algorithm

Arithmetic operator $LOG\nabla^2 g$ is forwarded by [7]. Two-dimensional LOG operator can be represented as:

$$\nabla^2 g = \left(\frac{x^2 - \sigma^2}{\sigma^4}\right) exp\left(-\frac{x^2}{2\sigma^2}\right) \otimes f(x, y) \quad (7)$$

Where $f(x, y)$ is the function to be detected. Using a typical template LOG operator in the text to detect whether the corner point measured under a certain specified scale level is the extreme value, which result in an invariant scale feature corner. The procedure of multi-scale Harris feature point detection algorithm procedure is as follows:

- Primarily select scale variables σ_n and the threshold value T, using formula (6) and (7) to calculate the candidate feature corner of each scale level.
- Use iterative algorithm to detect whether the LOG operator of each scale level candidate corner points to obtain the maximum value, and determine the results in the location and scale of the final feature corner. Consider the entire scale space of the image, assumed to detect the corner set C_{opt} under $\sigma_n = \sigma_0$ large-scale level. Decrease the image scale coordinate to $\sigma_n = \sigma_1$, and detect new corner set C_{new} in the neighborhood of the image. If there is C_{new}, regard C_{new} as the corner feature set of the current image. Repeat the above process until there is no change of C_{new}, or until the scale is small enough.

3) Compare Experimental Results of multi-scale Harris Feature Point Detection Algorithm with That of Ordinary Harris Algorithm

During experiment, taking the standard deviation proportional constant of Gaussian kernel function $S = 0.7$, $k = 0.04$ [7], use a typical 5×5 LOG operator.

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & -16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

 ISBN: 978-1-61208-158-8

When σ_n is taken as 2, 1 and multi-scale, the results of feature point extracting of the original image which is shown highlighted are shown as follows:

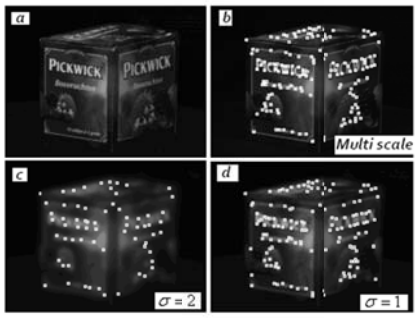


Figure1. Traditional Harris feature point detection of different scales compared with multi-scale Harris feature point detection

Stability criteria, reliability criteria, and anti-noise performance can be applied to evaluate the superior or inferior of the detection algorithm in a certain corner [8][12]. These three criteria are determined by the repetition rate γ of both initial detection of the corner and the corner points that detected after the change of parameters, threshold value, or the increase of the noise. In our experiment, we test the stability criteria of the multi-scale Harris feature point detection by changing the scare.

$$\gamma = \frac{|C_1 \cap C_2|}{min(|C_1|,|C_2|)} \times 100\% \quad (8)$$

In (8), C_i denotes the detected feature points set. $|C_i|$ represents the number of elements in the collection C_i. Formula (8) shows that the bigger the repetition rate γ is, the more stable the algorithm will be. Results of different scales for corner detection to the original image is showed in Figure 1, C_1 is the corner set with fixed scale of $\sigma = 1$, C_2 is the corner points set with the scale of $\sigma_n \in [2,6]$, calculate the repetition rate of C_1 and C_2 respectively. See the experimental results in Figure 2.

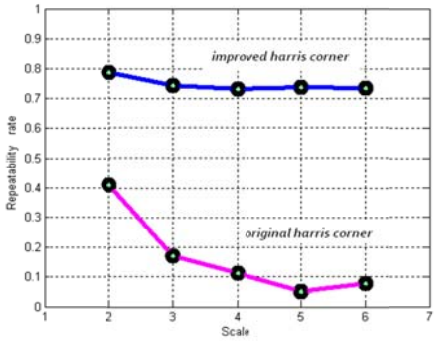


Figure 2. Comparison of 2 kinds Harris feature point repetition rate

It can be seen from the above graph that, when the scale changes, repeatability of the scale space theory which based on Harris feature point detection method for duplicate detection rate is significantly higher than the original Harris method, that is to say, Harris feature point detection method that based on scale space theory is more stable and reliable .

III. FEATURE POINT MATCHING

In this section, initial matching, epipolar geometry constraint and homography constraint are introduced. Two-way guided matching is proposed.

A. Initial Matching

Initial match consists of two steps: general matching and specific matching. General matching is achieved by respectively calculating the correlation coefficient of feature points of the two images. When the correlation coefficient is larger than a given threshold value, both the feature points are considered to be reciprocal, and therefore become the candidate of the corresponding feature points. x Is the feature point of I_1, while x' is the feature point of I_2. First of all, assume (x, x') is a real corresponding feature point, then within its neighborhood there might be more corresponding feature point (y, y'), where y and y' are respectively within the neighborhood of x and x'. The necessary condition that (y, y') is supported by real feature point (x, x') is: the angle betweenx, y and x', y' should be less than the preliminary set value. Search support strength from the neighborhood to accumulate matching strength. Since one feature point can match more than one candidate, each corresponding point has support strength. Therefore, we just consider the maximum support of each neighborhood and the symmetry of such support.

B. Using Epipolar Geometry and Homography Constraint to Eliminate Mismatches

In this section, fundamental matrix F and homography matrix H are calculated, and the mismatches are removed at the same time.

1) Epipolar Geometry Constraint

In two images that view from different view points, epipolar geometry constraint is the certain limit between the corresponding points of the same physical space point when collecting. It can be algebraically described by the fundamental matrix.

$$m'^T F m = 0 \quad (9)$$

In (9), F is fundamental matrix, m and m' is a pair of matching points of two images. Because there are many mismatches among the initial matching points, directly use of usual least squares method to calculate the fundamental matrix does not achieve good results. We choose the RANSAC [10][11] method, which is thought to be more robust. The process of using RANSAC method to get F is also the process of removing the mismatches.

2) Homography Constraint

Homography constraint maps the points of one geometry plane surface to another. It is a reversible mapping and obeys keeping linear. m and m' are corresponding match points of the two images, so that the homography H should obey:

$$m' = Hm \quad (10)$$

During experiment, we apply the RANSAC method again to find the solution for H, at the same time, further eliminate mismatches. The polar line runs through the entire image in epipolar geometry constraint, which still has relatively large

 ISBN: 978-1-61208-158-8

matching space. As for the images with lots of feature points, there are still plenty of mismatches after using epipolar geometry constraint. The mentioned homography constraint method above can further diminish the scope of matching so that matching precision is proved. H is a mapping of points from one geometric plane surface to another, which is inapplicable to the images of significant depth change. Nevertheless, most physical scene images have little depth change, by appropriately broadening the threshold of matching, good results can be achieved in practice.

3) Guided Matching by Using Epipolar Geometry and Homography Constraint

The above matching process has removed a large number of mismatches, but it gets less correct matching points at the same time. In order to solve this problem, we integrated the use of epipolar geometry and homography constraint to guided matching. Redirect and match all feature points of image I_1 and I_2 to get more accurate matching points.

The epipolar geometry constraint show that m_i' which is the matching point of m_i (m_i is the feature points detected in I_2) is necessarily adjacent to the corresponding polar line. According to homography constraint, m_i' is also near to estimation point $\hat{m}_i'$. S3 the intersection of the above two mentioned areas are the approximate area where the matching points located, see Figure 3below.

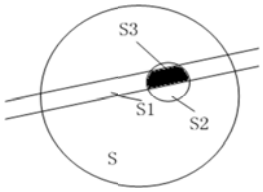


Figure 3. The searching scope of guided matching

Compared with the initial match, selecting the area defined in S3 to guided matching have greatly diminished the searching range of finding matching points and reduced but cannot avoid mismatching. Apply the euclidean distance detection method to calculate the euclidean distance between m_i' and $\hat{m}_i'$, sort the matching points from short to long in accordance with the euclidean distance, and select the top-ranked matching points. Further diminish the threshold for interior points of RANSAC method, and use top-ranked matching points to solve F, H. Then to guided match and select the accurate matching points with the euclidean distance again, see Figure 4, use two cycles guided matching to obtain more accurate F and H and make more accurate matching. The above steps have completed the guided matching of the feature points of I_1. Similarity, apply the same method to guided matching the feature points of I_2, and then take their union set.

Our experiment guides all the feature points for the second round of guided matching rather than keep only top-ranked points, because each round will get even better.

IV. EXPERIMENTS AND RESULT

The procedures and results of our experiment are shown in this section.

A. Experimental Operation

The entire process of experimental test operation is shown in Figure 4:

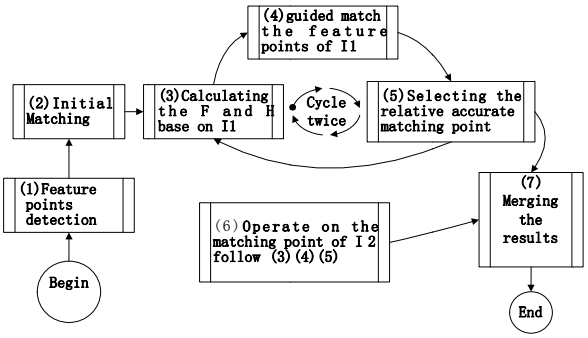


Figure 4. Flow chart of the whole experiment

B. Experimental Results

In order to verify the feasibility of proposed algorithm, this paper conducted experiments with two downloaded images captured by an un-calibrated SLR camera from different view points, and the parallax angle is 85 degrees. The two images we used are as follow:

Figure 5. Original images

The results are showed in Figure 6-Figure 8. Every pair of matching points in I_1 and I_2 is connected by a straight line in the figure. Figure 6 shows the results of initial matching, Figure 7 shows the matching results of epipolar geometry and homography constraint. Figure 8 shows the results of two-way guided matching. The number of accurate matching points before guided matching is 102, and the number of matching points after two-way guided matching is 320 pairs. We can see from the corresponding matching points from Figure 8 that the amount of the feature points is relatively large and match each other accurately, but the matching points which can express the features of the box, such as feature points at the top side of the box are not matched by our method for the epipolar geometry constraint can fit the whole image, but the homography constraint cannot. In our experiment, the method of WBM is even applicable to the image pairs whose parallax angle are 85 degrees, but it is more suitable to that less than 80 degrees. Figure 9-Figure 11 are the results of our approach by using the proposed images whose parallax angle is 65 degrees. We can easily find out that after applying two-way guided matching algorithm, nearly all feature points are matched correctly, and there is no error matching at all.

 ISBN: 978-1-61208-158-8

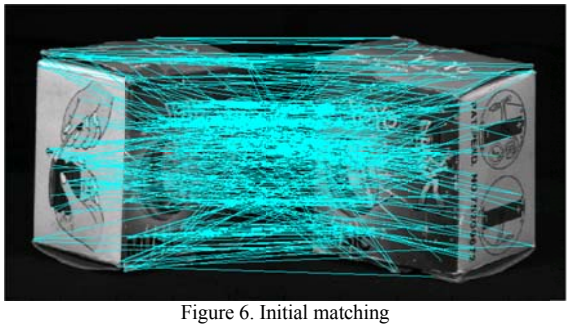


Figure 6. Initial matching

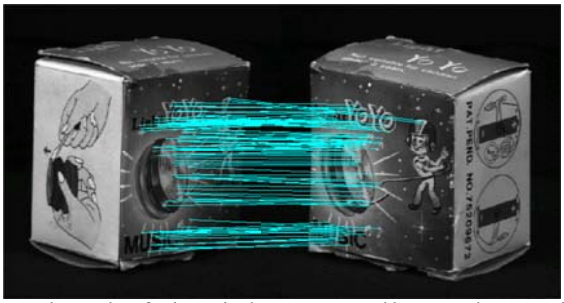


Figure 7. The results of using epipolar geometry and homography constraint to remove mismatches (102 pairs)

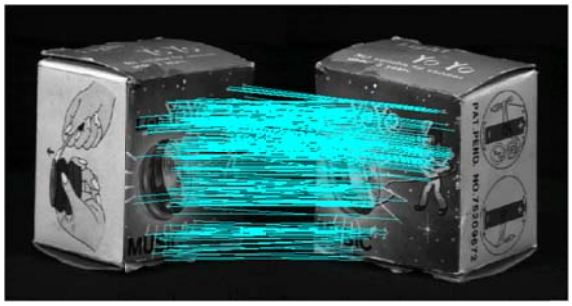


Figure 8. Two-way guided matching (320 pairs)

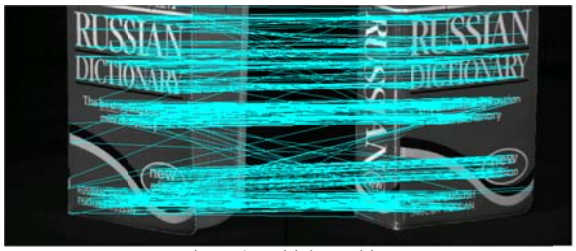


Figure 9. Initial matching

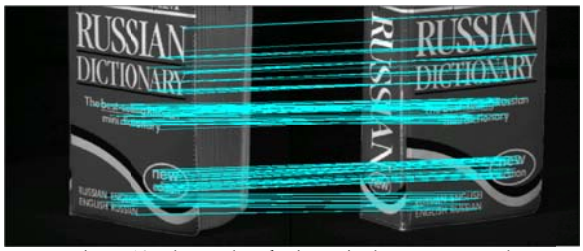


Figure 10. The results of using epipolar geometry and homography constraint to remove mismatches (198 pairs)

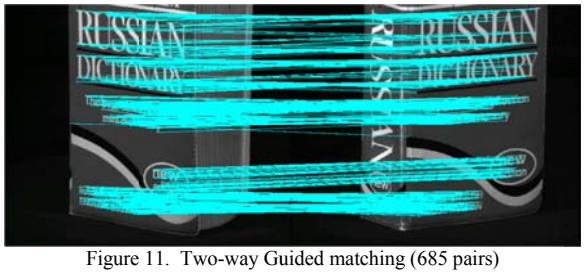


Figure 11. Two-way Guided matching (685 pairs)

To verify the correctness of our algorithm, we calculated the parallax angle of the images and conducted experiments on four groups of error rate testing. The results show that the calculated parallax angle is in accordance with the real parallax angle.

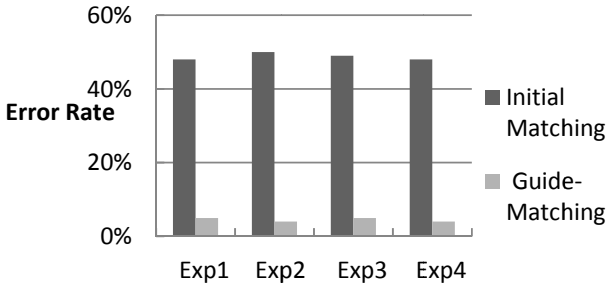


Figure 12. Results Statistics

Figure 12 shows the initial matching and the guided matching error rate. As can be seen from the chart, this method greatly reduces the matching error rate.

V. Conclusion and Future Work

In this paper, we successfully solved the problem of how to obtain sufficient and reliable matching points over two wide baseline image pairs. First, a large number of feature points are detected by using multi-scale Harris corner detection algorithm. Then, we comprehensively apply epipolar geometry and homography constraint to guided matching algorithm, which effectively settle the conflict between wide baseline and sufficient matching points without reducing the accuracy of the matching points. Meanwhile we have applied Euclidean distance to filter the matching points, making the matching more precise. Finally, we have tested a number of wide baseline image pairs under different severe camera motions with illumination changes, self-similarities, and have obtained the excellent results for all of the images. The experiments show that this algorithm can effectively detect and match, and it also has better match property compared with the traditional corner detection and matching algorithm. This method can increase the quantity of common points in 3D and then enhance the accuracy of 3D reconstruction. However, due to the homography constraint of geometric plane surface that used in this algorithm, the pictures with great depth change usually get poor test results, which still needs to be improved in future research.

 ISBN: 978-1-61208-158-8

VI. ACKNOWLEDGEMENT

This paper is supported by the Fundamental Research Funds for the Central Universities, Number: 1107021051, and National Natural Science Fundation of Jiangsu Province of China, Number: BK2010386.

REFERENCES

[1] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions via graph cuts", Proceedings of International Conference on Computer Vision, 2001, pp. 508–515.

[2] O. Chum and J. Matas. "Matching with prosac" Progressive sample consensus". Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 220–226.

[3] P. Pritchett and A. Zisserman., "Matching and reconstruction from widely separated views", Proceedings of the European Workshop on 3D Structure from Multiple Images of Large-Scale Environments, 1998, pp. 78–92.

[4] C. Wang, K.K. Ma, T. Khim, and D. Guo, "Mismatch Removal for Wide baseline Image Matching via Coherent Region-to- Region", Proceedings of Fourth Pacific-Rim Symposium on Image and Video Technology, 2010, pp. 101-106

[5] C. Schimid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors", International Journal of Computer Vision, 2000 37(2), pp. 151-172

[6] K. Peng, X. Chen, D. Zhou, and Y.H. Liu, "3D Reconstruction Based on SIFT and Harris Feature Points", Proceedings of International Conference on Robotics and Biomimetics, 2009, pp. 960-964

[7] C. Harris and M. Stephens. "A combined corner and edge detector", Proceedings of 4th Alvery Vision Conference, 1988, pp. 147-151

[8] D. Lowe. "Distinctive Image Features from Scale-Invariant Interest Points". International Journal of Computer Vision 2004, 60(2), pp. 91-110

[9] Z. Zhang, R. Deriche, and O. Faugeras, "A robust technique for matching two uncalibrated images trought the unknown epipolar geometry". Artificial Intelligence, Special volume on computer vision, 1995, 78(1-2), pp. 87-119

[10] L. Tony. "Seale-space A framework for handing image structures at multiple Seales". Proceedings of the conference Eurean organization for Nuclear Research school of computing, Egmond aan zee, The Netherland, 1996, pp. 8-21.

[11] L. Tony. "Scale-space for Discrete signals". IEEE transactions on pattem Analysis and machine intelligence, 1990, pp. 233-253

[12] Y. Yang and T.W. Zhang. "Assessing criterion of corner finders" Journal of Harbin Institute of Technology, 1998, 30(2), pp. 7-10.

[13] R. Hartley and A. Zisserman, "Multiple view geometry in computer vision". Cambridge University Press, 2000, pp. 237-259.

[14] Lowe and G. David G "Object recognition from local scale-invariant features". Proceedings of the International Conference on Computer Vision. 2. pp. 1150–1157

 ISBN: 978-1-61208-158-8

Boosted Wireless Capsule Endoscopy Frames Classification

Giovanni Gallo, Alessandro Torrisi
Dipartimento di Matematica e Informatica
Università di Catania
Catania, Italy
Email: {gallo, atorrisi}@dmi.unict.it

***Abstract*—Intestinal lumen detection in endoscopic images is clinically relevant to assist the medical expert to study intestinal motility. Wireless Capsule Endoscopy (WCE) produces a high number of frames and automatic classification, indexation and annotation of WCE videos is crucial to a more widespread use of this diagnostic tool. In this paper we propose a novel intestinal lumen detection method based on boosting. In particular, we use a customized set of Haar-like features combined with a variant of Adaboost to select discriminative features and to combine them into a cascade of strong classifiers. Experimental results show the efficacy of boosted classifiers to quickly recognize the presence of intestinal lumen frames in a video.**

***Keywords*-Classification; Pattern Recognition; Boosting; Wireless Capsule Endoscopy; Video Automatic Annotation.**

I. Introduction

Wireless Capsule Endoscopy [1], [2] is a technique to explore small intestine regions that traditional endoscopy does not reach. A video-capsule, that integrates wireless transmission with image technology, is swallowed by the patient and it is propelled through the gut by intestinal peristalsis. Once activated, the capsule captures two frames per second and transmits images to an external receiver. The exam is concluded after about eight hours, that corresponds to the lifetime of the battery of the capsule. Images taken during the entire route of the capsule through the intestine are successively analyzed by an expert. She may spend up to one or more hours to gather the relevant information for a proper diagnosis. This greatly limits the use of the capsule as a diagnotic routine tool.

This state of the things may be greatly improved if the WCE video is automatically segmented into shorter videos, each one relative to a different trait of the bowels, and if reliable automatic annotation tools are available. The goal of automatically produce a summary of the whole WCE video is still unaccomplished. Tools to extract semantic information from such videos are hence relevant.

Within this general framework, in this paper we present a novel method to automatically discriminate a relevant subclass of frames. In particular, our classifier sorts the frames into two categories: "with lumen" (images depicting the stages of a intestinal contraction where the lumen of the bowel is well visible) and "without lumen" (Figure 1). Lumen detection is clinically relevant because it announces the presence of a contraction and helps the physician to study intestinal motility. Alteration of physiological intestinal motility is an indicator of disorders in which the gut has lost its ability because of endogenous or exogenous causes. Lumen detection in our approach is obtained as a special case of object detection, and uses to this aim the Viola and Jones paradigm introduced in 2001 [3].

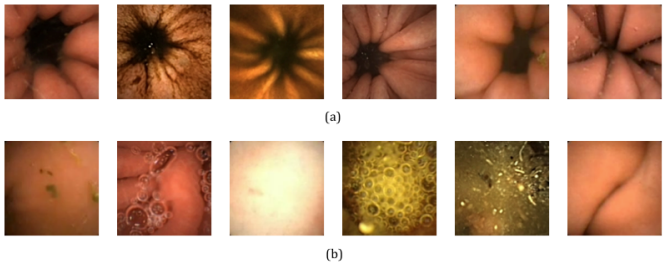


Figure 1. Examples of *lumen* (a) and *not lumen* (b) images extracted from a WCE video.

This paper is organized as follows: Section II lists related works and reports examples of object detection based on an approach similar to ours. Section III describes in detail how Viola-Jones technique is applied to the present problem. Section IV reports the experiments conducted on real WCE video. Finally, Section V draws conclusions and discusses some future works.

II. Related Works

Most of the systems reported in literature to recognize intestinal lumen images refer to classic endoscopy. The motivation behind these methods is to aid the physician to individuate lumen region to avoid the collision of the endoscope instrument with the intestinal mucosa. In this context, Asari [4] proposes a Region Growing Segmentation to extract lumen from gray level endoscopic images.

Recently, the original WCE is evolving into a novel capsule whose movement may be remotely controlled. In this context, the recognition of lumen could help the capsule to go through the intestine minimizing collisions and avoiding to record meaningless frames. To this aim, Zabulis et al. [5] propose a system based on a Mean Shift Segmentation algorithm variant to locate lumen regions in WCE frames. The problem of the detection of images with lumen in WCE

 ISBN: 978-1-61208-158-8

videos to clinical use is not much investigated. Some works study the general problem to find contractions to examine intestinal motility [6]–[8].

The main idea exploited in this work is the Viola-Jones method for object detection [3], [9]. Initially proposed for face detection, this technique is based on the use of simple features calculated in a new representation of the image. Based on the concept of integral image [10], a huge set of features is tested and the boosting algorithm Adaboost is used to reduce this set [11]–[13]. The introduction of a tree of boosted classifiers provides a robust and fast detection and minimizes the false positive rate. This strategy has been proven effective to recognize various kind of objects. Several systems have been proposed for different recognition problems, like face, hands and pedestrian [14]–[17]. The possibility to define a specific set of features and the more recent release of an open source implementation [18] have permitted to use extensively this method in Computer Vision.

III. Proposed Method

We propose a system that automatically learns and classifies frames where intestinal lumen is well visible. The process of automatic annotation of images containing intestinal lumen can be summarized in the following three steps:

- Evaluation of a customized set of Haar features applied to the integral images of the training samples.
- Selection of best discriminative features through AdaBoost algorithm.
- Construction of a final boosted classifier based on a cascade of classifiers whose complexity is gradually increasing.

To be reliable, such a system must satisfy two requirements: it needs a comprehensive set of examples where the object of interest may occur; a suitable selection of descriptors required to describe each possible pattern must be guaranteed. To this aim, Haar-like features, a set derived from Haar wavelets [19], recognize objects using intensity contrast between adjacent regions in an image.

Basic Haar features proposed by Viola-Jones for face detection do not have sufficient discriminative power for lumen investigation: it necessary to define customized variations of this kind of features. In particular, the features proposed in this work should provide a strong positive response on a rectangular region with low intensity called generically "lumen" and a brighter surrounding area in which appears the gut wall. By combining a learned evaluation threshold to each feature, it is possible to assign an image to the appropriate category. Figure 2 shows an example of this first kind of proposed features that we call "center-surround" features.

The typical appearance of a frame that shows an intestinal contractions consists in lumen surrounded by typical rays that muscular tone produces due to the folding of the intestinal wall. We hence define two additional "cross-like" kind

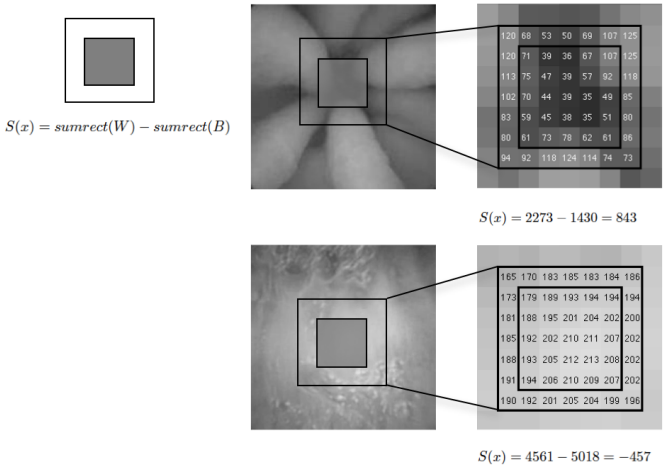


Figure 2. Evaluation of a "center-surround" feature in two images, with and without lumen respectively. $sumrect$ indicates the total value of pixels intensity within a rectangular region. W and B are related to light and dark regions of the rectangle.

of features that enhance the discriminative power produced by the simpler "center-surround" feature (Figure 3). The calculation of this second kind of features may be efficiently obtained as for the simpler "center-surround" features.

Using integral image representation, feature evaluation is accomplished by few memory accesses. It is straightforward to verify that to compute "center-surround" features, at any position or scale, only eight lookups are needed. The remaining two feature typologies require more accesses due to greater number of rectangular areas. "Cross-features" require respectively 16 and 24 references from the integral image.

Once a feature shape has been assigned, it is necessary to specify the position and the scale within the region of interest. Specifically, the features are scanned across the image top left to bottom right using a sliding offset of two pixels both in the horizontal and in the vertical direction. The process is iteratively repeated with different feature scales at each round. To keep the computation of the proposed features within the same number of lookups into the integral image, we choose not to change the scale of the image but to variate instead the size of the features.

The exact representation for the three proposed types of features is as follows:

$$f = [x_w, y_w, s_{wx}, s_{wy}, x_b, y_b, s_{bx}, s_{by}, type, \theta, \rho] \qquad (1)$$

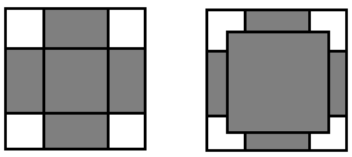


Figure 3. The two "cross-like" patterns used in the proposed method.

 ISBN: 978-1-61208-158-8

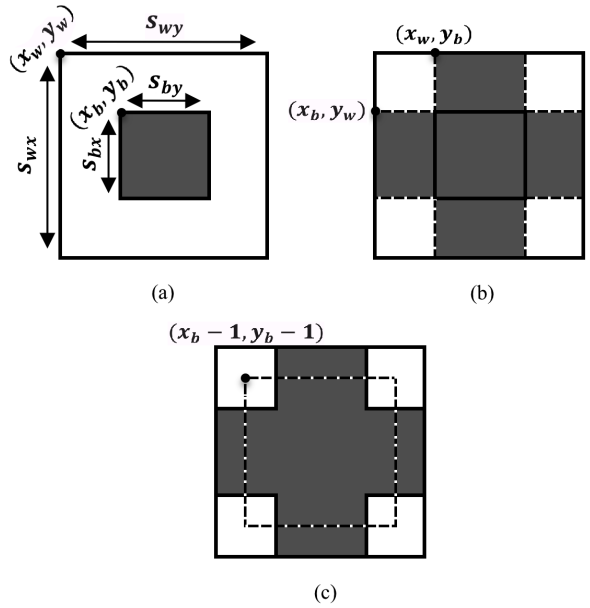


Figure 4. Schematic features representation. (a) Center-surround feature. (b) First cross feature obtained by center-surround feature considering the cross with width s_{by} and height s_{bx}. (c) Second cross feature obtained by the first taking into account a inner square of width and height greater than 1 pixel respect to the previous version.

The first four elements of f refer to the larger square of the feature. Similarly, the following four elements relate to the inner square. The $type$ parameter is an integer that indicates which type of feature is considered. The last two parameters are the optimal learned threshold and the polarity to register the category of images discriminated by that feature.

As discussed before, the "center-surround" features are evaluated considering difference between the sum of the pixels within two rectangular regions (Figure 4a). The second type of features considers a cross-shaped region to enhance lumen area. Location and size of this region are constrained by the size of correlated "center-surround" feature (Figure 4b). The third type of features is calculated in a similar way considering a larger internal rectangular area (Figure 4c). We consider the same total number of features for each type.

We consider only squared features, i.e., those with equal horizontal and vertical scale s_w. The internal region relative to lumen varies from a minimum size 2×2 up to $(s_w - 2) \times (s_w - 2)$ pixels. Once fixed the size of the external section, the descriptor associated with the lumen is shifted across the external descriptor resizing at each step (Figure 5).

The resolution of a WCE frame is reduced to 24×24 pixels in this phase of processing. Hence the total number of features per scale is equal to the total amount of different features in the image multiplied by the allowed variations of magnitude. For example, a 7×7 feature contains sixteen regions of size 2×2, nine of size 3×3, four of size 4×4 and one of size 5×5. Hence, the total number of features

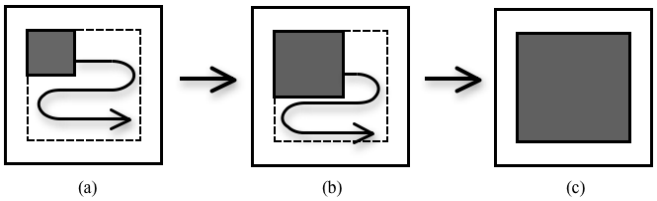


Figure 5. An example of features variations used in our algorithm. Given feature size, all 2×2 regions are considered in each location (a). This cycle is reiterated by increasing the size of the inner square (b) until maximum amplitude is achieved (c).

of size 7×7 is 2430, equal to the number of windows in the image (assuming a horizontal and vertical offset of two pixels) for the total number of variations. Table I summarizes the feature counting for the chosen scales.

A. Training

During the training phase, the dataset is rescaled to 24×24 pixels. The integral image representation of gray tone training samples is used to compute feature scores. Application of AdaBoost provides a list of best discriminative features. In particular, we build a binary classifier for each feature (weak classifier). Initially all the examples have the same weight. For each boosting step, the determination of a new weak classifier involves the evaluation of each feature on training data. The crucial feature is selected according to the weighted error that each feature shows on the training data. In the successive round, the samples are reweighted to emphasize the misclassified ones. This is the most expensive section of the training module.

The result of the training module is a strong classifier computed as weighted linear combination of weak classifiers built during each round of boosting. The whole boosting process is iterated, varying at each step the number of weak classifiers. The result is the realization of a cascade of strong classifiers with a gradually increasing number of features.

Learning requires that each strong classifier shows a prescribed detection rate, while maintaining a definite rate

Table I
FEATURES NUMBER PER SCALE. THE FIRST COLUMN REFERS TO THE SIZE OF THE FEATURE WHILE THE SECOND IS RELATED TO MAXIMUM SCALE ALLOWED FOR THE LUMEN AREA.

$Feature\ size$	$Max\ Internal\ scale$	$\#Features$	$\#Variations$	$Total$
5×5	3×3	100	5	500
7×7	5×5	81	30	2430
9×9	7×7	64	91	5824
11×11	9×9	49	204	9996
13×13	11×11	36	385	13860
15×15	13×13	25	650	16250
17×17	15×15	16	1015	16240
19×19	17×17	9	1496	13464
21×21	19×19	4	2109	8436

 ISBN: 978-1-61208-158-8

of false positives. In particular, it is required a minimum detection rate and a maximum false positive rate to be satisfied at every level of the cascade. For each strong classifier, a weak classifier is added until it reaches the required parameters for the current level of the cascade. Similarly, a new strong classifier is associate to the cascade until total false positive rate is above a certain threshold.

One of the advantages of the proposed system is that the user only needs to define the features set to be used and the false positives and detection rates for each level of the cascade. All the internal parameters are automatically selected during the training phase.

B. Test

In the proposed system, each test image is scaled to 24×24 pixels and it is labelled as "lumen" or "not lumen". This monoscale procedure combined with selection of best features during training allows real time application of our system. Please notice that, differently than in the case where the object to recognize may appear at different scales, in the present case a "mono-scale" choice has been shown adequate.

IV. Experimental Results

In this section, we report of the experiments carried out to verify the efficacy of the proposed method. To train the classifier, we have considered the integral images of a dataset made of 5000 images, 1500 positive and 3500 negative, rescaled to 24×24 pixels. The positive images have been manually selected from WCE videos previously labelled by the expert. The selected images represent a comprehensive set of scenes where the intestinal lumen can be present, including location and scale changes within the image. Differently, the negative examples have been randomly selected from videos that not contain any lumen. Both typical smooth images and images containing other judged negative events, like the presence of bubbles, bleedings, residuals, share this set.

To train the cascade of classifiers, we need to establish a maximum false positive rate and a minimum detection rate to satisfy to each layer of cascade. In particular, we require that 98% of positive images must be recognized at each level while maintaining a maximum amount of false positives equivalent to 80%. At the next levels of the cascade these two values are computed relatively to the new dataset whose positives set is composed by every lumen recognized as such by the previous classifier and the negatives set includes the remaining false positives. A strong classifier will be added to the cascade until the total false positive rate drops to zero. With these data, we have obtained a six level cascade for a total of 325 features (Figure 6). The total detection rate of the cascade, D, and the final false positive rate F, are obtained as a combination of intermediate outcomes on the cascade:

dataset	1500 L 3500 NL	***dataset***	1475 L 1965 NL	***dataset***	1451 L 861 NL
d_1	98,33%	d_2	98,37%	d_3	99,65%
f_1	56,14%	f_2	43,81%	f_3	34,95%
features	3	***features***	19	***features***	75
1° *Strong classifier*		**2° *Strong classifier***		**3° *Strong classifier***	
dataset	1446 L 301 NL	***dataset***	1436 L 83 NL	***dataset***	1436 L 1 NL
d_4	99,30%	d_5	100%	d_6	100%
f_4	27,57%	f_5	1,2%	f_6	0%
features	76	***features***	76	***features***	76
4° *Strong classifier*		**5 *Strong classifier***		**6° *Strong classifier***	

Figure 6. Cascade of classifiers. d_i and f_i represent detection and false positive rate at the ith level of cascade. L and NL indicate "lumen" and "not lumen" images, respectively.

$$D = \prod_{i=1}^{N} d_i = 95,7\% \qquad F = \prod_{i=1}^{N} f_i = 0\% \qquad (2)$$

where N is the total number of layers of the cascade.

To test the effectiveness of trained cascade, we have considered a collection of ~ 5000 images randomly extracted from a test set of frames disjoined from the training set. During testing phase, we consider the integral images of test set rescaled to 24×24 pixels with the respective labels, the cascade of boosted classifiers as it has been obtained during training and, finally, a threshold that determines the rigorousness of the classifier. Each test sample gets through each single node of the cascade; a positive outcome is sent by the classifier i to the more complex classifier $i + 1$. An image is labeled as lumen if positively overcomes each node of the cascade. If at any point the test image is judged negative, it is rejected immediately without further test. The classification performance has been evaluated in terms of precision and recall by comparing our results with the annotations provided by the specialist. Table II shows the results.

All experiments have been conducted on a consumer level PC with Intel®Core™2 Duo processor and 4 GB of RAM. Calculations have been performed in MATLAB environment.

By varying the rigidity threshold from a minimum to a maximum value, we can construct a ROC curve comparing the detection rate versus the number of false positives.

Table II
Classification results

Recall	*Precision*	*Accuracy*	*Time*
$90,5\%$	71%	$92,4\%(4642/5025)$	$600 frames/sec$

Higher threshold values minimizes both detection and false positive rates. Similarly, a low threshold will lead to acceptance of a greater number of lumen images while increasing the probability of detecting false positives. Figure 7 reveals that is possible to reach a detection rate above 90%, keeping the amount of false positives at about ~ 500 instances, i.e., 10% of the test dataset.

Figure 8 shows some examples of the false positives for the proposed method. In many circumstances, the intensity contrast between adjacent regions does not correspond to the presence of a lumen. A common problem is the fact that Haar features are sensitive to illumination changes. Variations on the lighting conditions may cause the cascade to detect lumen that was not predicted during the training stage. Likewise, in some images, folds of the intestinal wall may produce contrasted regions that confuse the Haar features. If new kind of images are presented to the classifier, detection is difficult and the amount of false positives increases. To deal with this problem, training data must include as many examples as possible to predict only true lumen.

V. Conclusion

In this paper we introduced an automatic lumen detection algorithm for endoscopic images. Inspired by Viola-Jones object detection system, we show that using AdaBoost learning-based algorithm combined with a cascade of strong classifiers leads to a good rate of detection minimizing running time. Experimental results show that the proposed system detects positive images using exclusively Haar-like proposed features. A cascade of six strong classifiers reaches a recall of 90,5% with a precision of 71%. Our detector is flexible and easily extensible to other semantic objects in endoscopic applications.

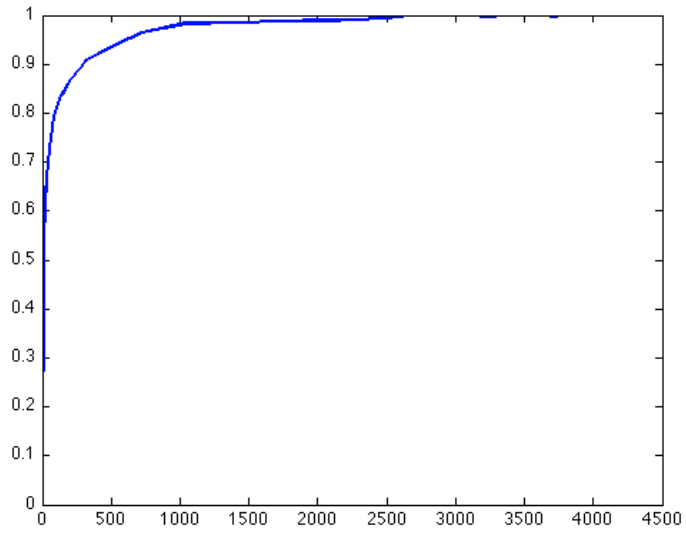


Figure 7. ROC curve for the detector with 325 weak classifiers.

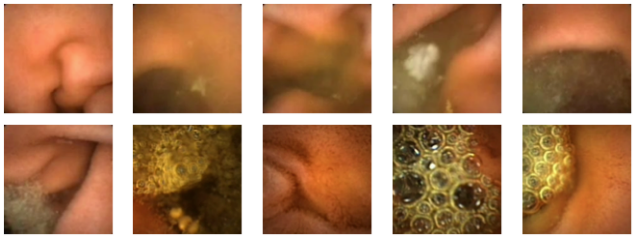


Figure 8. Example of some false positives detected by the system.

References

[1] G. Imaging, "Expanding the scope of gi," Last accessed: July 2011. [Online]. Available: http://www.givenimaging.com

[2] G. Iddan, A. Glukhovsky, and P. Swain, "Wireless capsule endoscopy," *Nature*, vol. 405, pp. 725–729, 2000.

[3] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, 2001, pp. 511–518.

[4] K. Asari, "A fast and accurate segmentation technique for the extraction of gastrointestinal lumen from endoscopic images," in *Medical Engineering and Physics*, vol. 22, 2000, pp. 89–96.

[5] X. Zabulis, A. Argyros, and D. Tsakiris, "Lumen detection for capsule endoscopy," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, sept. 2008, pp. 3921–3926.

[6] P. Spyridonos, F. Vilarino, J. Vitria, F. Azpiroz, and P. Radeva, "Anisotropic feature extraction from endoluminal images for detection of intestinal contractions," in *Medical Image Computing and Computer-Assisted Intervention*, 2006, pp. 161–168.

[7] G. Gallo and E. Granata, "Lbp based detection of intestinal motility in wce images," vol. 7961, no. 1. SPIE, 2011, p. 79614T. [Online]. Available: http://link.aip.org/link/?PSI/7961/79614T/1

[8] J. Lee, J. Oh, S. K. Shah, X. Yuan, and S. J. Tang, "Automatic classification of digestive organs in wireless capsule endoscopy videos," in *Proceedings of the 2007 ACM symposium on Applied computing*, ser. SAC '07. New York, NY, USA: ACM, 2007, pp. 1041–1045. [Online]. Available: http://doi.acm.org/10.1145/1244002.1244230

[9] P. Viola and M. Jones, "Robust real-time object detection," in *International Journal of Computer Vision*, 2001.

[10] F. C. Crow, "Summed-area tables for texture mapping," in *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '84. New York, NY, USA: ACM, 1984, pp. 207–212. [Online]. Available: http://doi.acm.org/10.1145/800031.808600

 ISBN: 978-1-61208-158-8

[11] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998. [Online]. Available: http://dx.doi.org/10.2307/120016

[12] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *Proceedings of the Second European Conference on Computational Learning Theory*. London, UK: Springer-Verlag, 1995, pp. 23–37. [Online]. Available: http://portal.acm.org/citation.cfm?id=646943.712093

[13] Y. Freund and R. Schapire, "A short introduction to boosting," *J. Japan. Soc. for Artif. Intel.*, vol. 14, no. 5, pp. 771–780, 1999. [Online]. Available: citeseer.ist.psu.edu/freund99short.html

[14] P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, vol. 2, oct. 2003, pp. 734–741.

[15] L. Yun and Z. Peng, "An automatic hand gesture recognition system based on viola-jones method and svms," in *Computer Science and Engineering, 2009. WCSE '09. Second International Workshop on*, vol. 2, oct. 2009, pp. 72–76.

[16] M. Kolsch and M. Turk, "Analysis of rotational robustness of hand detection with a viola-jones detector," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 3, aug. 2004, pp. 107 – 110 Vol.3.

[17] M. Castrillon-santana, O. Deniz-suarez, L. Anton-canals, and J. Lorenzo-navarro, "Face and facial feature detection evaluation performance evaluation of public domain haar detectors for face and facial feature detection," 2008.

[18] OpenCV, "Open computer vision library," Last accessed: July 2011. [Online]. Available: http://opencv.willowgarage.com

[19] C. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Computer Vision, 1998. Sixth International Conference on*, jan 1998, pp. 555 –562.

 ISBN: 978-1-61208-158-8

Improved Trend Following Trading Model by Recalling Past Strategies in Derivatives Market

Simon Fong, Jackie Tai
Department of Computer and Information Science
University of Macau
Macau SAR
ccfong@umac.mo, ma56562@umac.mo

***Abstract*—Unlike financial forecasting, trend following (TF) doesn't predict any market movement; instead it identifies a trend at early time of the day, and trades automatically afterwards by a pre-defined strategy regardless of the moving market directions during run time. Trend following trading has a long and successful history among speculators. The traditional TF trading method is by human judgment in setting the rules (aka the strategy). Subsequently the TF strategy is executed in pure objective operational manner. Finding the correct strategy at the beginning is crucial in TF. This usually involves human intervention in first identifying a trend, and configuring when to place an order and close it out, when certain conditions are met. In this paper, we proposed a Trend Recalling model that operates in a computer system. It works by partially matching the current trend with one of the proven successful patterns from the past. Our experiments based on real stock market data show that this method has an edge over the other trend following methods in profitability. The results show that TF however is still limited by market fluctuation (volatility), and the ability to identify trend signal.**

Keywords- Trend Following; Automatic Trading System; Futures Contracts; Mechanical Trading

I. INTRODUCTION

Trend following (TF) [1] is a reactive trading method in response to the real-time market situation; it does neither price forecasting nor predicting any market movement. Once a trend is identified, it activates the trading rules and adheres rigidly to the rules until the next prominent trend is identified. Trend following does not guarantee profit every time, but nonetheless in a long term period it may probably profit by obtaining more gains than loses. Since TF is an objective mechanism that is totally free from human judgment and technical forecasting, the trends and patterns of the underlying data play an absolutely influential role in deciding its ultimate performance.

It was already shown in [2] that market fluctuation adversely affects the performance of TF. Financial cycles are inevitable phenomena and it is a controversy whether cycles can be predicted or past values cannot forecast future values because they are random in nature. Nonetheless, we observed that cycles could not be easily predicted, but the abstract patterns of such cycles can be practically recalled by simple pattern matching. The formal interpretation of financial cycle (or better known as economic cycle) refers to economy-wide fluctuations in production or economic activity over several months or years. Here we consider it as the cycle that run continuously between bull market and bear market, some people refer this as market cycle (although they are highly correlated), In general a cycle is made of four stages, and these four stages are: "(1) consolidation (2) upward advancement (3) culmination (4) decline" [3]. Despite being termed as cycles, they do not follow a mechanical or predictable periodic pattern. But similar patterns are observed to always repeat themselves in the future, just as a question of when, though in approximate shapes. We can anticipate that some exceptional peak (or other particular pattern) of the market trend that happen today, will one day happen again, just like how it did happen in history. For instance, in the "1997 Asian Financial Crisis" [4], the Hang Seng Index plunged from the top to bottom (in stages 3 to 4); then about ten years later, the scenario repeats itself in the "2008 Financial Crisis" [5].

Dow Theory [6] describes the market trend (part of the cycle) as three types of movement. (1) The "primary movement", main movement or primary trend, which can last from a few months to several years. It can be either a bullish or bearish market trend. (2) The "secondary movement", medium trend or intermediate reaction may last from ten days to three months. (3) The "minor movement" or daily swing varies from hours to a day. Primary trend is a part of the cycle, which consist of one or several intermediate reaction and the daily swings are the minor movements that consist of all the detailed movements. Now if we project the previous assumption that the cycle is ever continuously rolling, into the minor daily movement, can we assume the trend that happens today, may also appear some days later in the future?

Figure 1. Intra-day of 2009-12-07 and 2008-01-31 day trend graphs

 ISBN: 978-1-61208-158-8

Here is an example for this assumption; Figure 1 shows respectively two intra-day 2009-12-07 (top) and 2008-01-31 (bottom) trend graphs of Hang Seng Index Futures, which are sourced from two different dates. Although they are not exactly the same, in terms of major upwards and downwards trends the two graphs do look alike.

This is the underlying concept of our recalling trading strategies that are based on searching for similar patterns from the past. This concept is valid for TF because TF works by smoothing out the averages of the time series. Minor fluctuations or jitters along the trend are averaged out. This is important because TF is known to work well on major trending cycles aka major outlines of the market trend.

II. RECALLING PAST TRENDS

An improved version of Trend Following algorithm is proposed in this paper, which looks back to the past for reference of selecting the best trading strategy. The design of a TF system is grounded on the rules that are summarized by Michael W. Covel, into the following five questions [7]:

1. How does the system determine what market to buy or sell at any time?
2. How does the system determine how much of a market to buy or sell at any time?
3. How does the system determine when you buy or sell a market?
4. How does the system determine when you get out of a losing position?
5. How does the system determine when you get out of a winning position?

The first and second questions are already answered in our previous work [1]. The third question is rather challenging, that is actually the core decision maker in the TF system and where the key factor in making profit is; questions 4 and 5 are related to it. Suppose that we have found a way to identify trend signal (to buy or sell), and we have a position opened. Now if the system along the way identifies another trend signal, which complies with the current opened position direction, then we should keep it open, since it suggested that the trend is not yet over. However, if it is counter to the current position, we should probably get a close out, regardless whether you are currently winning or losing, as it indicates a trend reversion.

Our improved TF algorithm is designed to answer this question: when to buy or sell. It is a fact that financial cycles do exist, and it is hypothesized that a trend on a particular day from the past could happen again some days later. This assumption supports the Recalling trading mechanism, which is the basic driving force that our improved trend following algorithm relies on. The idea is expressed in Figure 2. As it can be seen in the diagram there's four major processes for the decision making. Namely they are Pre-processing, Selection, Verification and Analysis. Figure 2 shows the process of which our improved TF model works by recalling a trading strategy that used to perform well in the past by matching the current shape of the pattern to that of the old time. A handful of such patterns and corresponding trading strategies are short-listed; one strategy is picked from the list after thorough verification and analysis.

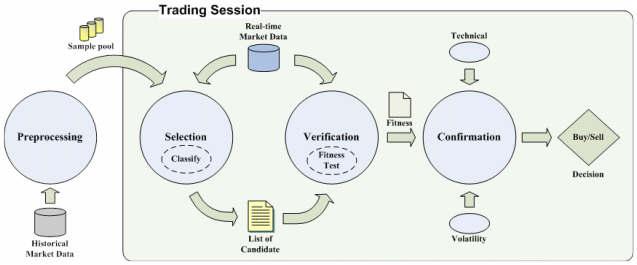


Figure 2. Improved TF process with Recalling function.

A. *Pre-processing*

In this step, raw historical data that are collected from the past market are archived into a pool of samples. A sample is a day trend from the past with the corresponding trading strategy attached. The trend is like an index pattern for locating the winning trading strategy that is in the format of a sequence of buy and sell decisions. Good trading strategy is one that used to maximize profit in the past given the specific market trend pattern. This past pattern which is deemed to be similar to the current market trend, is now serving as a guidance to locate the strategy to be applied for decision making during the current market trade session.

Since the past day trend that yielded a great profit before, reusing it almost can guarantee a perfect trading strategy that is superior than human judgment or a complex time series forecasting algorithm. The past samples are referenced by best trading strategies on an indicator that we name it as "*EDM*" (exponential divergence in movement). *EDM* is a crisp value indicator that is based on two moving average differences.

$$EDM_{(t)} = f\left(EMA_{s(t)} - EMA_{l(t)}\right),$$

$$EMA_{(t)} = \left(price_{(t)} - EMA_{(t-1)} \times \frac{2}{n+1} \right) + EMA_{(t-1)}$$

where $price(t)$ is the current price at any given time t, n is the number of periods, s denotes a shorter period of $EMA(t)$ at time t, l represents a longer period $EMA(t)$, $f(.)$ is a function for generating the crisp result. The indicator sculpts the trend; and base on this information, a TF program finds a list of best trading strategies, which can potentially generate high profit. The following diagram is an example of pre-processing a 2009-12-07 day trend that shows the *EDM*. As indicated from the diagram the program first found a long position at 10:00 followed by a short position at around 10:25, then a long position at 11:25, finally a short position around 13:51 and closes it out at the end of the day, which reaps a total of 634 index points. In Hong Kong stock market, there is a two hours break between morning and afternoon sessions. To avoid this discontinuation on the chart, we shift the time backward, and joined these two sessions into one, so 13:15 is equivalent to 15:15.

B. *Selection*

Once a pool of samples reached a substantial size, the improved TF with recalling mechanism is ready to use. The stored past samples are searched and the matching ones are selected. The goal of this selection process is to find the most

 ISBN: 978-1-61208-158-8

similar samples from the pool, which will be used as a guideline in the forthcoming trading session. A foremost technical challenge is that no two trends are exactly the same, as they do differ day by day as the market fluctuates in all different manners. Secondly, even two sample day trends look similar but their price ranges can usually be quite different. With consideration of these challenges, it implies that the sample cannot be compared directly value to value for a simple match. Some normalization is necessary for enabling some rough matches. Furthermore the comparison should allow certain level of fuzziness. Hence each sample trend should be converted into a normalized graph, and by comparing their rough edges and measure the difference, it is possible to quantitatively derive a numeric list of similarities. In pattern recognition, the shape of an image can be converted to an outline like a wire-frame by using some image processing algorithm. The same type of algorithm is used here for extracting features from the trend line samples for quick comparison during a TF trading process.

In our methodology, each sample is first converted into a normalized graph, by calculating their technical indicators data. Indicators such as RSI and STC have a limited value range (from 1 to 100) which is suitable for fast comparison, and they are sufficient to reflect the shape of a trend. In other words, these indicators help to normalize each trend sample into a simple 2D line graph. We can then simply compare each of their graphical difference by superimposing these line graphs on top of each other for estimating the differences. This approach produces a hierarchical similarity list, such that we can get around with the inexact matching problem and allows a certain level of fuzziness without losing their similarity attributes. Figure 4 shows an example of two similar samples trend graphs with the RSI displayed.

Figure 3. Example of EDM and preprocessed trend of 2009-12-17

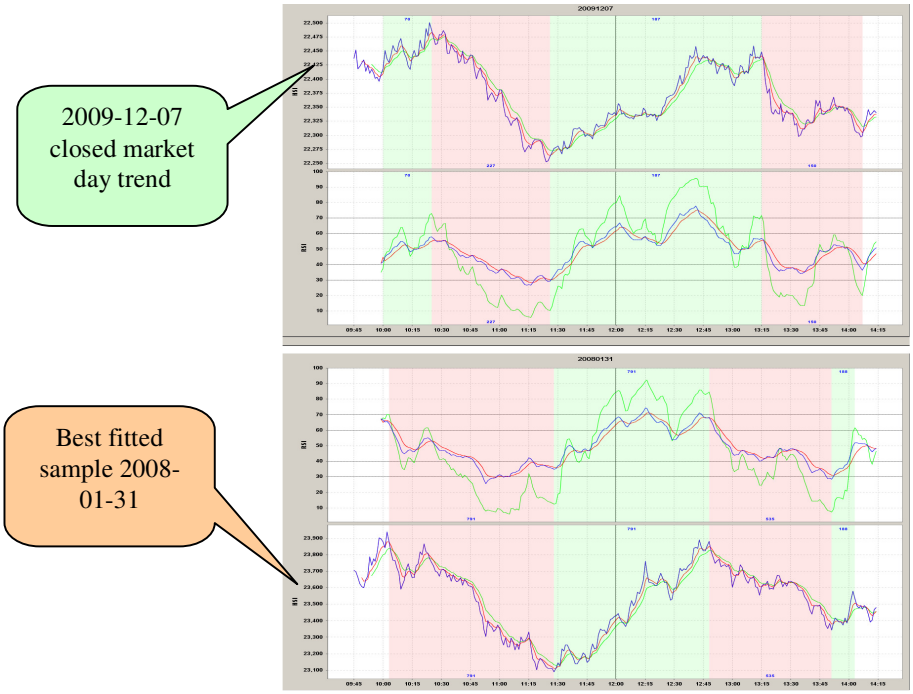


Figure 4. Example of 2009-12-07 sample with best fitness (2008-01-31) day trend and RSI graph.

 ISBN: 978-1-61208-158-8

C. *Verification*

During this process, each candidate from the list will be tested against the current market state. Ranking from the top smallest number as the most similar, they will be passed through fitness test. Each trend sample is corresponding to a specific trading strategy (that was already established in the pre-processing step). Each trading strategy will be extracted and evaluated against historical data and calculating how well it performed as a trial. Each of their performances will be recorded. The trial performance will be used as a criterion to rearrange the list. Here we have an example before and after the fitness test, which was run on the 2009-12-07 during the middle of simulated trade session.

List of Candidate	
Rank	Sample
0	**2008-01-25**
1	2008-03-07
2	2009-08-18
3	2008-01-23
4	2008-11-10
5	2007-08-29
6	2008-04-16
7	2007-09-13
8	2008-12-15
9	2008-02-01
10	2009-11-25
11	2008-01-31
12	2008-02-21
14	2007-12-05
15	**2009-03-17**

List After Fitness Test		
Sample	Fitness	Rank
2009-03-17	**373**	**15**
2007-09-13	357	7
2008-12-15	341	8
2008-02-01	339	9
2008-01-31	326	11
2007-12-05	305	14
2008-01-25	**300**	**0**
2008-02-21	296	12
2009-08-18	294	2
2008-01-23	273	3
2008-11-10	270	4
2007-08-29	179	5
2009-11-25	165	10
2008-03-07	143	1
2008-04-16	3	6

Figure 5. Fitness test applied on 2009-12-07 at the time 14:47

Verification is needed because the selection of these candidates is by a best effort approach. That is because the market situations now and the past may still differ to certain extent.

D. *Confirmation*

After the verification process is finished, the candidate list is re-sorted according to the fitness test results. The fittest one will be used as the reference of subsequent trading strategy during the TF decision making. In order to further improve the performance on top of the referencing to the past best strategy, some technical analysis is suggested to be referenced as well. By the advice of Richard L. Weissman from his book [8], the two-moving average crossover system should be used as a signal confirmation. (The two-moving average crossover system entails the rise of a second, shorter-term moving average.) But instead of using simple moving average, EMA - exponential moving average with RSI should be used, that is a short-term RSI EMA and a long-term RSI EMA crossover system. When a trend is identified and it appears as a good trading signal, the crossover system must also be referenced and check if it gives a consistent signal. Otherwise the potential trend is considered as a false signal or noise. For example in our case the trading strategy from the recalled sample hints a long position trade. We check if RSI crossover system shows a short-term EMA crossing over its long-term EMA or not.

In addition to validating the hinted trading signals from past strategy, market volatility should be considered during decision making. There are many ways to calculate volatility; the most common one is finding the standard deviation of an asset closing price over a year. The central concept of volatility is finding the amount of changes or variances in the price of the asset over a period of time. So, we can measure market volatility simply by the following equation:

$$Volatility_{(t)} = SMA_n\left(\left(\ln(price_{(t)}) - \ln(price_{(t-1)})\right) \times C\right),$$

$$SMA_{(t)} = \frac{Close_{(t)} + Close_{(t-1)} + ...Close_{(t-n+1)}}{n}$$

where ln(.) is a natural logarithm, n is the number of periods, t is the current time, C is a constant that enlarges the digit to a significant figure. By observing how the equation responds to historical data, we can find the maximum volatility as ±15. Base on the previous fluctuation test result, we can define it as the following fuzzy membership.

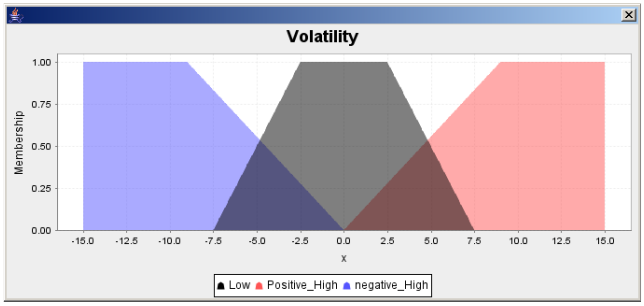


Figure 6. Volatility membership defination.

During the trading session, volatility will be constantly referenced while the following rules apply at the TF system:

IF volatility is too positive high and long position is opened THEN close it out

IF volatility is too positive high and no position is opened THEN open short position

IF volatility is too low THEN do nothing

IF volatility is too negative high and short position is opened THEN close it out

IF volatility is too negative high and no position is opened THEN open long position

These rules have a higher priority over the trade strategies, such that when the condition has met any of these rules, it will take over the control regardless of what decision that the trade strategies has made. We now summarize these four processes as pseudo codes shown in Appendix.

III. EXPERIMENT

The improved TF algorithm with recalling function is programmed as an automated trading simulator, written in JAVA. All trials of simulations are run on historical market data within 2.5 years prior to the year of 2010. A price that spread between bid and ask price is considered on each trade.

 ISBN: 978-1-61208-158-8

In the simulation each trade is calculated in the unit of index point, each index point is equivalent to 50 HKD, which is subject to overhead cost as defined by Interactive Broker unbundled commission scheme at 19.3 HKD per trade. ROI is the prime performance index that is based on Hong Kong Exchange current Initial margin requirement (each contract 7400 HKD in year of 2010).

From the simulation result, we observed as shown in Table 1 that a 333% ROI is achieved at the end of the experimental run. This is a significant result as it implies the proposed TF algorithm can reap more than three folds of whatever the initial investment is over just 2.5 years of trading. The trading pattern is shown in Figure 7 that shows overall the trading strategy is ever winning in a long run. Figure 8 shows a horizon of trading results, overall there are more profits than losses.

TABLE I. RESULTS OF THE OVERALL SIMULATION RUNS

Simulation	STF
Total Index Point	5867
Net Worth (HKD)	293350
Total Trade	1216
Cost (HKD)	46938
P&L (HKD)	246412
ROI (Year)	333%

IV. CONCLUSION AND FUTURE WORKS

Trend following has been known as a rational stock trading technique that just rides on the market trends with some preset rules for deciding when to buy or sell. TF has been widely used in industries, but none of it was studied academically in computer science communities. We pioneered in formulating TF into algorithms and evaluating their performance. Our previous work has shown that its performance suffers when the market fluctuates in large extents. In this paper, we extended the original TF algorithm by adding a market trend recalling function. Trading strategy that used to make profit from the past was recalled for serving as a reference for the current trading. The trading strategy was recalled by matching the current market trend with the past market trend at which good profit was made by the strategy. Matching market trend patterns was not easy because patterns can be quite different in details, and the problem was overcome in this paper. Our simulation showed that the improved TF model is able to generate profit from stock market trading at more than three times of ROI. Our next step is to test the algorithms on other stock market data.

REFERENCES

[1] Fong S. and Tai J., "The Application of Trend Following Strategies in Stock Market Trading", The 5th International Conference on Networked Computing, 25-27 August 2009, Seoul, Korea, pp. 1971-1976.

[2] Fong S., Tai J., and Si Y.W., "Trend Following Algorithms for Technical Trading in Stock Market", Journal of Emerging Technologies in Web Intelligence (JETWI), Academy Publisher, ISSN 1798-0461, Volume 3, Issue 2, May 2011, Oulu, Finland, pp. 136-145.

[3] Stan Weinstein's, Secrets for Profiting in Bull and Bear Markets, pp. 31-44, McGraw-Hill, USA, 1988.

[4] Wikipedia, 1997 Asian Financial Crisis, Available at http://en.wikipedia.org/wiki/1997_Asian_Financial_Crisis, accessed Sep-2010.

[5] Wikipedia, Financial crisis of 2007–2010, Available at http://en.wikipedia.org/wiki/Financial_crisis_of_2007, accessed Sep-2010.

[6] Schannep J., "Dow Theory for the 21st Century: Technical Indicators for Improving Your Investment Results", Wiley, USA, 2008.

[7] Covel M.W., "Trend Following: How Great Traders Make Millions in Up or Down Markets", New Expanded Edition, Prentice Hall, USA, 2007, pp. 220-231.

[8] Weissman R.L., "Mechanical Trading Systems: Pairing Trader Psychology with Technical Analysis", Wiley, USA, 2004, pp. 10-19.

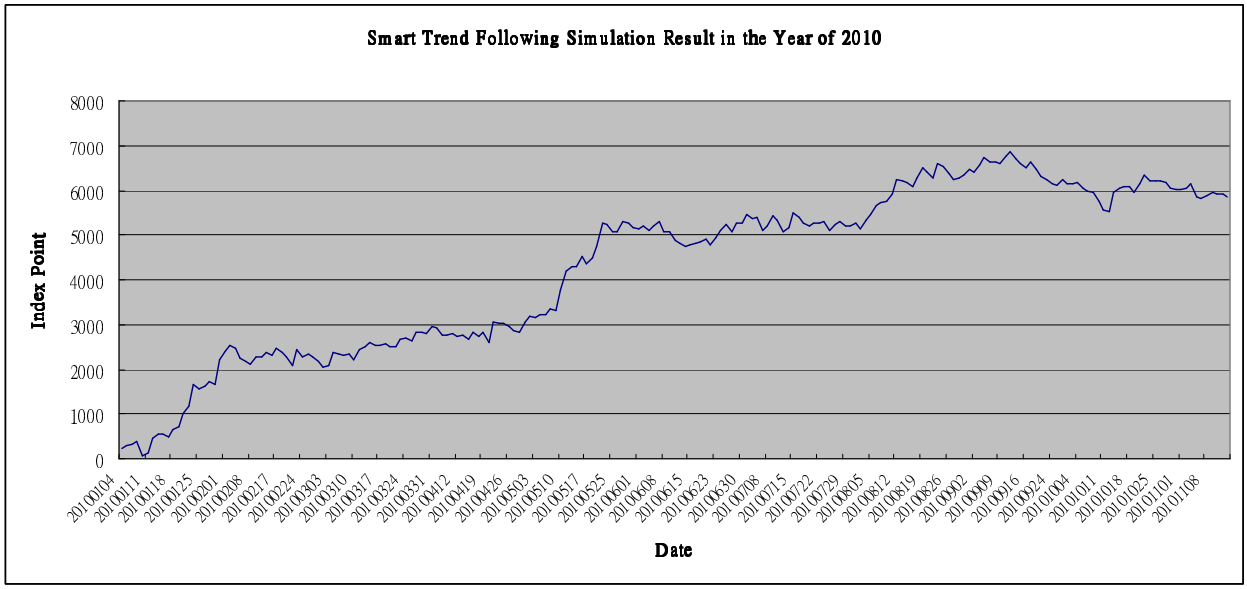


Figure 7. Performance of the improved TF algorithm with recalling function

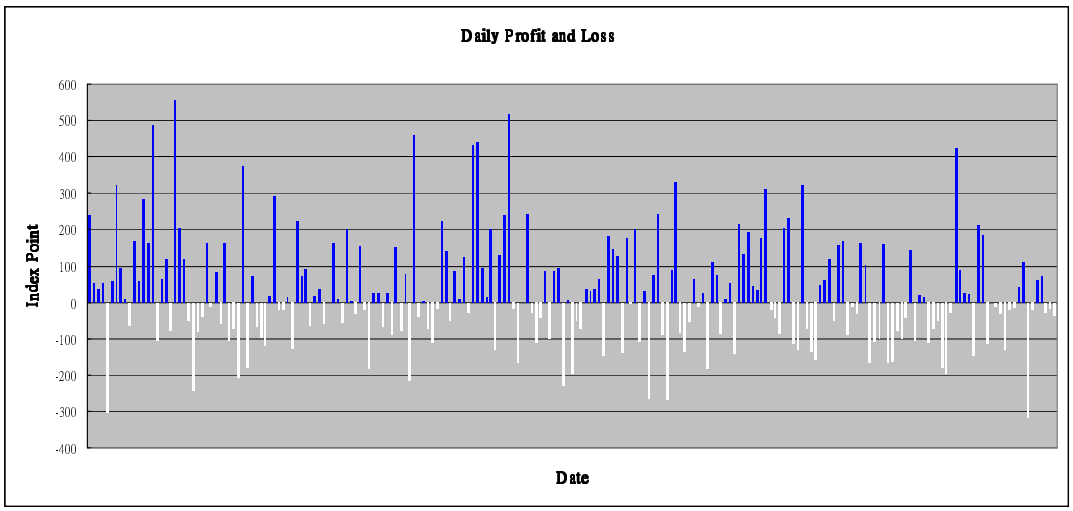


Figure 8. Daily profit and loss diagram.

Appendix

```
Loop all historical data
    Loop each minute within each day
        Compute and save technical data
        Compute EDM

        Found all the turning point according to EDM
        For each two connected turning point
            Found their respective position
            Calculate P&L according to the position
            If P&L is positive
                Save position and each point time line
            Else
                Adjust each point time line
        End-loop
    End-loop
End-loop
```

```
Loop each minute until end of market
    Compute technical data
    Compute volatility

    For each sample in pool
        Compare sample technical data with current market technical data
        Save their similarity into list
    End-loop

    Sort the list with most similar on the top

    For 1 To 20 in the list
        Apply fitness test on each sample with current market trend
        Save their performance into list
    End-loop

    Sort the list with most fitness on the top

    Extract strategies from the top fitness sample

    Reference RSI crossover system

    Apply fuzzy sets

    If fuzzy sets not fire
        If no position is opened
            If strategy shows long position and confirm with RSI crossover
                Open long position
            Else if strategy shows short position and confirm with RSI crossover
                Open short position
        Else if long position is opened
            if strategy shows short position and confirm with RSI crossover
                Close long position
                Open short position
        Else if short position is opened
            if strategy shows long position and confirm with RSI crossover
                Close short position
                Open long position

    If end of the day
        Close any opened position
End-loop
```

Visualizing Multiple Websites with Views of Structural Growth, Dynamic Performance and Peer Comparison

Simon Fong, Ho Si Meng, Ma Cheng I
Department of Computer and Information Science
University of Macau
Macau SAR
ccfong@umac.mo, smho@dsej.gov.mo, iris.ma@venetian.com.mo

***Abstract*—Website visualization is an important research topic in data analytics that has its practical applications in performance management reports and navigational problem diagnosis. Researchers from their previous works studied different forms of visualization mainly as 3D linked graphs. In this paper, we propose an innovative tree model that represents website structures as radial and concentric circles of nodes, and users' interactions with the websites as staggered threads. These two types of visualization, when used together, effectively express the growth of a website in a static view and the usage of its services in a dynamic view respectively. Furthermore, the proposed techniques could be placed collectively together for visualizing websites comparatively across different countries or regions as an ecosystem of forest of trees. This paper presents this new concept as well as the features of the custom-made simulator programmed in Java.**

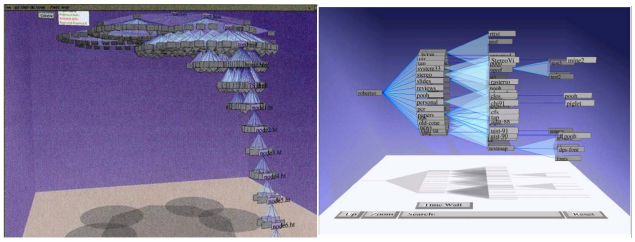
Keywords-Website visualization; Web ecosystem.

I. INTRODUCTION

Website visualization refers to some graphical representation of website information such as the hierarchical structures, the usage data and the natures of the contents etc. The graphical display, which visually conveys a conceivable kind of relations among the data, helps the user to build a mental model about the websites. This mental model is supposedly more effective to be understood about the insights of the website, than the traditional lengthy reports and/or simple displays of tree structures like site-map.

Tree layouts have a long history in visualizing websites as node-link diagrams. Nodes that represent webpages are directly linked with edges that represent the hyperlinks. Web structure visualization in form of tree layouts has been a popular application in evaluating web usability [1]. The tree graph visually displays the possible click-paths through which users can navigate along a website. Another application is to monitor the growth of a website, by visually checking over the document structure and the amount of the contents in a form of tree, ensuring that the growth in all directions is balanced. A balanced growth means information is evenly distributed across different sections and pages of the website for an all-rounded development.

In the pioneer works [2], websites were commonly represented by Cone Tree and Kam Tree, in Figure 1 (a) and (b). Cone Tree has a root node placed at the apex of the cone dangling from it are the child nodes that lined up uniformly. Cone Trees usually are in 3D, so viewing the child nodes on the other side of the baseline requires spinning the tree like a carousal. It is difficult for the child nodes to be labeled in a Cone Tree. To overcome this difficulty Kam Trees were later invented where the root node is on the most left of the graph and the child nodes are stretching across to the right. Focus+context technique was proposed too for making use of inherent perspective projection in the 3D environment. Cone and Kam Trees are known to be ideal for visualizing some medium sized website of around 100 notes, with more of a focus view on a specific piece of information on the tree. Any detail of a tree can be faithfully displayed. They are however not suitable for visualizing an overall website structure because a compact and zoom-out view may be preferred over a potentially very large number of nodes of a website.



(a) Cone Tree (b) Kam Tree

(c) Hyperbolic Browsers

Figure 1. (a) Cone Tree - Courtesy of katsiafas.com; (b) Kam Tree – Courtesy of ifs.tuwien.ac.at; (c) Examples of Hyperbolic Browser [4]

Hyperbolic Browser [3], in a nutshell, maps a tree structure of any size within a bounded circle so that the whole Euclidean plane can always be viewed entirely. In other words, a large number of nodes (hence a large website) can be represented in a single snapshot of hyperbolic display. Relational information among the nodes and links can be shown through coloring or label placement schemes. Hyperbolic Browser still is limited by space constraints.

 ISBN: 978-1-61208-158-8

II. WEBSITE STRUCTURE AS RADIAL TREE PATTERN

In this paper, we extended the hyperbolic tree representation into a radial tree pattern by adding in different colors and sizes of the tree nodes that cluster to a common parent node. The concept derives naturally from the hierarchy of a site-map where any parent node of the tree embraces a cluster of child nodes, and child nodes of the child nodes across several subordinate levels. All these nodes would be visualized collectively by the same color originated from the topmost parent node. In this case, colors represent different major categories of webpages in a website. This is important for website evaluation because the distribution of the colors indicates whether the major categories of website grow in balance or not. Furthermore, the sizes of the nodes symbolize the amounts of information carried by the corresponding web pages. The combination of colors and sizes allows a user to visually check if the website has any biased proportion of information (in terms of weights represented by sizes of nodes) and number of web pages (in terms of spread of colors represented by the numbers of the nodes). Any too heavily biased part of the website can be early identified and the development could be subsequently rectified. Moreover the same radial tree pattern can be used to show the popularity of each section or each page of the website, for the radial tree pattern essentially is representing the whole website structure. Popularity in the context of a website is expressed as the number of hits or page views on each page. Similar to the concept of *heat map*, hot spots of the website can be shown in different colors. When these two modes are used together, managers can evaluate how much resources are invested in various parts of the website contents and how well they are favored by frequent visits.

Figure 2. Radial Tree Patterns; Above – the pattern is showing the static structure of the website with different weights at each categories of pages; Below – the same pattern now uses coloring to show the popularity of each part of the website.

This visualization scheme would be particularly useful for organization or a government that wants to evaluate a collection of their websites vis-à-vis. For example, an e-government portal usually consists of many departmental websites, each of which operates as an individual site that has its own team to update the information and used by different segments of citizens. In this case, multiple websites could be monitored comparatively as a row of radial tree patterns. At one glance, the user can spot the strength or weakness of each site from observing over its structure and weights of contents, as well as popularity. If the patterns are collected over a long time, the growth of these websites can be longitudinally studied and visually compared too.

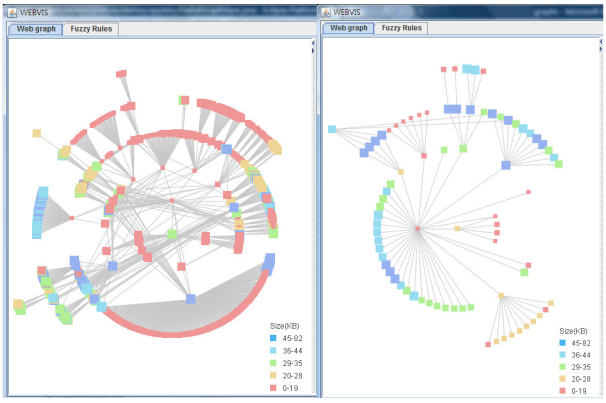


Figure 3. Visual comparison of Radial Tree Patterns of two websites; Left – The website of Education and Youth Affairs Bureau of Macao showing complex and large structure with six levels. It contains more than 900 pages. The node color is indicating the page size. Most pages are in pink having little content.; Right – The website of Fundo de Segurança Social, Macau with simple and small structure having five levels. It contains nearly 100 pages. Few pages but heavy in contents.

The Radial Tree Patterns are technically programmed in Java with a visualization toolkit called Prefuse. A prototype is implemented from our previous project WebVS [5].

III. WEBSITE VISITS AS FUZZY RULES PATTERN

While Radial Tree Patterns are useful for illustrating the full view of a website and the information pertained at different parts of the website, they fall short in visualizing the association of parts of the websites being visited. This association is also known as clickstream in Web mining where a temporal sequence of visits over a series of webpages frequently occurred. Many research papers in the literature [6] have been focusing on this problem. Visualization of such associations is implemented here along with Radial Tree Patterns because dynamic usages of the website that are represented by web visits complement with the growth of website structures and contents as a holistic approach. Importantly, this holistic approach inspires website visualization to a higher level – collective and comparative visualization of websites as a forest of trees. Essentially the information from the growth of website structure and from the dynamic usages of the website serves as input values for modeling parameters of a forest of tree (FOT). A sample of the fuzzy association rules from WebVS is shown in Fig. 4.

 ISBN: 978-1-61208-158-8

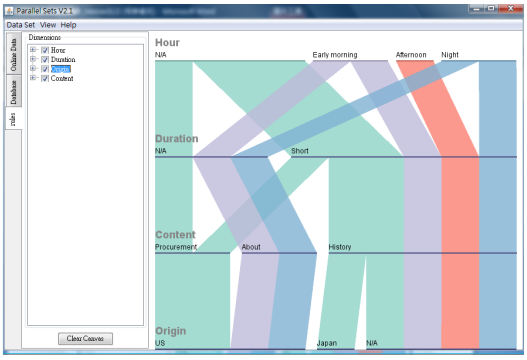


Figure 4. Visualizaton of fuzzy association rules that illustrate which parts of the website are often visited together at different time periods.

As a metaphor, the growth of a website structure is similar to the growth of a biological tree, that spans outwards of their branches and leaves. The longer the establishment of a website in history, the taller and bigger the representative tree is. Metaphorically speaking again, the dynamic usage of a website is analogous to its vital power that promotes the growth the website, hence its representative tree growth. This is based on the assumption that organizations or companies would only invest on the website expansion provided that the website maintains its popularity and thus its worthiness. And the investment is proportional to the rating of popularity of the website measured by hits and page views. Likewise, websites will be left dormant if few people are using them, so are their representative trees withering for a slow death.

IV. FOREST OF TREES FOR WEBSITES COMPARISON

Given the information from static website structures and dynamic usage, which in turn have already been visualized, an innovative modeling scheme of a forest of trees is proposed in this paper. The performance information from a website are mapped to the growth of a tree that represents the website. Table 1 below shows the basic parameters used in modeling FOT. The visualization program is custom developed by using Java and SExI-FS (Spatially Explicit Individual-based Forest Simulator, in full name) from World Agroforestry Center.

TABLE I. MODELING PARAMETERS IN SEXI-FS

SExI-FS parameters	Mapping of web performance indices to the modeling parameters
DBH (Diameter at Breast Height)	Height = Age (history of website in years) DBH = Logarithmic function of (general performance of the website as measured by users feedback/satisfaction, and/or web visits)
Crown Porosity	Amount of information on the website
Survival Probability	Growth = DBH performance. If growth = 0 or below a threshold, survival probability = sustainability of cost for maintaining the website
Mortality Modifier	Economic factors that affect the web operations
Secondary Mortality Probability	Political factors that affect the growth of the websites
Light Sensitivity	Minimum Light Level: The min level for a tree so that the tree can be grown -default 0.15
Imperata Competition Factor	Dynamic usage of the websites in relation to the rival websites

The modeling of the FOT is geographically based, where one tree and its position represent a particular website respectively and the trees are established over different countries or regions. The prime usefulness of visualizing a FOT is for comparing websites of different regions in the perception of different trees in a forest. Tall and strong trees symbolize prosperous websites that are developing well as shown by their well appearance, and they are receiving abundant nutrients for growth by the high usage of the users.

For an example of comparing e-Government websites over an Asia-Pacific region, as shown in Figure 5, the representative trees are growing at their respective countries and they take different shapes and sizes. Some e-Government websites, as presented by their trees are tall (because of long history) but relatively thin in the branches and leaves. That means these websites though have a long history of establishment, their service functions and contents are relatively sparse, which is typical for developing countries. In contrast, some modern cities have their e-Government websites represented by sturdy tree with lush leaves – they are the websites with good functionalities, rich information and popularly used by their citizens.

The FOT is still in progress of being developed, as many technical challenges still imposed. They are basically the detailed mapping of the performance variables from the dynamic usage and association rules of the website. In an abstract level, mapping the structure and simple well-being variable of a tree from the Radial Tree Pattern to a cone tree in FOT is straightforward. However, when the whole modeling is put to function as an ecosystem, many dependent variables exist and they need to be precisely calculated and mapped from the detailed operational variables (KPI's) of a website.

V. CONCLUSION

Website visualization has long been studied by researchers via a variety of visualization models from 3D to hyperbolic browsers. The purpose of website visualization traditionally is limited to allow interactive inspection of navigation paths of the site as represented by nodes and links in the graph. In this paper we proposed a new model that serves slightly different purposes for website visualization. Radial Tree Patterns were used to show the integrity of the website structures, as well as the popular spots of the website by allowing users to visualize the nodes in different colors and sizes. While colors represent homogeneity of categories of web pages, size of the nodes reveal how much contents there are in the webpages. The extent of the balanced growth of a website can be observed from the tree pattern, as well as the distribution of popularity received by the web visitors.

Instead of visualizing a website individually, a novel simulation model is proposed and formulated for comparing multiple websites as a forest; each tree represents a website in a specific region. The trees of the forest are visually compared in terms of their appearances which are derived from performance variables. For future work, we will fine-tune the mappings of the variables from individual trees to FOTs. The forest will be simulated as an ecosystem that depicts competitions and dependencies among the websites.

 ISBN: 978-1-61208-158-8

REFERENCES

[1] Chi Ed., Pirolli P., and Pitkow J., "The Scent of a Site: A System for Analyzing and Predicting Information Scent, Usage, and Usability of a Web Site", Proceedings of the SIGCHI conference on Human factors in computing systems, volume 2, Issue 1, April 2000, pp. 161-168.

[2] Robertson G.G., Card S.K., and Mackinlay J.D., "Information Visualization Using 3D Interactive Animation", Comm. ACM, vol. 36, no. 4, 1993, pp. 57-71.

[3] Lamping J. and Rao R., "Laying Out and Visualizing Large Trees Using a Hyperbolic Space", Proc. 7th Symp. User Interface Software and Technology (UIST 94), ACM Press, New York, 1994.

[4] Neumann P., "Focus+Context Visualization of Relations in Hierarchical Data", Diplom Thesis, Faculty of Computer Science, Universität Magdeburg, Germany, September 6, 2004, pp. 25.

[5] Ho S.M. and Fong S., "Visualizing e-Government Portal and Its Performance in WEBVS", The 5th International Conference on Digital Information Management (ICDIM 2010), July 2010, Thunder Bay, Canada, pp. 315-320.

[6] Baldini P. and Giudici P., "Improving Web Clickstream Analysis: Markov Chains Models and Genmax Algorithms", Mathematical Methods for Knowledge Discovery and Data Mining, IGI, 2008, pp. 233-243.

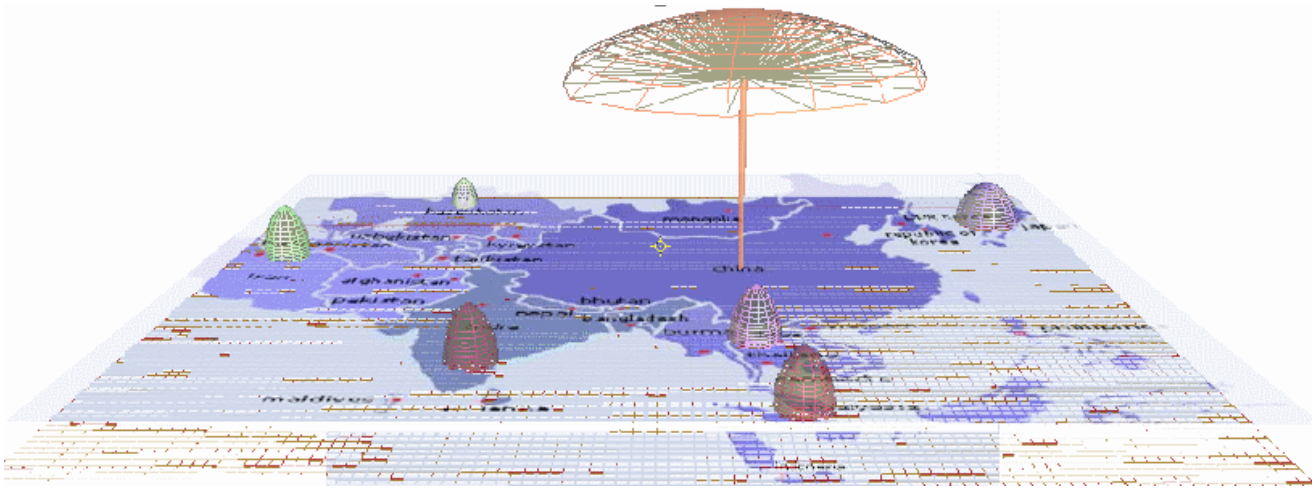


Figure 5. An example of the Forest of Tree visualized by SExI-FS. The graphic shows different e-Government websites are visualized as trees with different shapes and sizes.

Figure 6. Top down view of the Forest of Trees, where the crown of the tree represents the population of the users the website is supposed to serve.

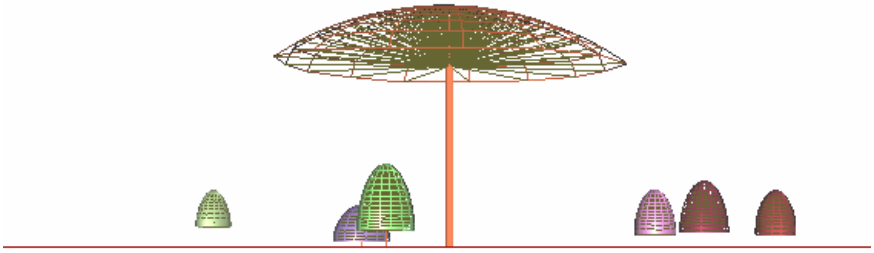


Figure 7. Side view of the Forest of Trees, where the height of the tree trunk represents the age of the website, the sizes of the branches denote the number of services or website categories are available and the density of the leaves represents the amount of information on those webpages.

 ISBN: 978-1-61208-158-8

Application of Motion Vector in Live 3D Object Reconstruction

Renshu Gu, Jie Yuan, Sidan Du
School of Electronic Science and Engineering, Nanjing University
Nanjing, China
Email: gu_renshu@yahoo.cn, yuanjie@nju.edu.cn, coff128@nju.edu.cn

***Abstract*—This paper applied a novel method to live 3D object reconstruction. Provided that static 3D reconstruction has been completed with basic frames, the proposed linear method calculates free 3D motion of a rigid-body in a video sequence, so as to display the moving object. The succinct method does not involve the fundamental matrix, and the point correspondence procedure in 3D reconstruction is reused. Least Mean Square decomposition is adopted to solve linear equation set and thus obtain motion vector. Furthermore, an iterative process enables the method to eliminate outliers. Sufficient experiments proved the validity and efficiency of the method.**

Keywords-3D motion; volumetric display; point correspondence; linear equation set; QR decomposition.

I. INTRODUCTION

Estimating rigid-body 3-D motion parameters from two or more images of an image sequence has been extensively studied for a long time in the field of computer vision. It can be tracked down to the 1970s, when early researches proved that three views are necessary to recover motion from an orthographic camera projection model, and that two views are enough to estimate motion from full perspective projection [1].

In 1989, Weng, Huang and Ahuja [2] gave a detailed discussion on motion estimation from two views of full perspective projection, and proposed the classic 8-point algorithm, namely to utilize the epipolar geometric constraint, estimate the basic matrix from feature points (matrix E in [2]), and then obtain 3D motion parameters from the basic matrix. From then on, to overcome its sensitivity to noise, a large number of improved methods were proposed, including American researcher Hartley's Improved 8-point Algorithm which standardizes 2D data. Except from methods based on point feature, researchers also proposed various methods based on line feature [4] [5], point feature and line feature combined, optic flow [6] [7] and methods using multiple frames and iterative algorithm [8] [9] [10].

Nevertheless, the methods mentioned above are integrated methods of 3-D motion estimation from two (or more) images without any prior knowledge of the object. In general, how to apply the epipolar geometric constraint is a key problem. As for methods based on point feature, recent works focus on error analysis and control (e.g., [1] [11] [12]), or on better estimation of the fundamental matrix [13] [14] [15], all of which inevitably engage the fundamental matrix. The most recent and relevant work in [16] described 2-frame recovery of structure and motion using uncalibrated cameras. This is somewhat similar to Han-Kanade's method in [17].

To perform live 3D object reconstruction, we intend to display moving 3D object in accordance with real pictures from live videos. However, adopting Han-Kanade reconstruction to video sequences over time would be too time-consuming. In our scheme, static 3D reconstruction is performed only at intervals, while motion vector is calculated and added to the static model at any time during the intervals. This paper proposed a linear method to calculate motion alone instead of both motion and structure, added the motion to the original 3D object, and then displayed it from whatever view the spectator prefer.

Once static 3D reconstruction has been completed, we know the 3D coordinates corresponding to 2D feature points in image t0, and we also know new 2D coordinates of those points in image t1 through image matching. It is convenient for us to match images, since it is a step in 3D reconstruction. With these data we know, 3D motion parameters can be calculated already, and there is no need to use the integrated methods in the above articles. In other words, we have obtained enough information of 2D mapping 3D object so there is no need for the epipolar constraint. In theory, with 3D reconstruction completed, 3 point pairs from 2 views can recover 3D motion when motion is small. This paper gives Approximate Algorithm applicable to small motion, and also Universal Algorithm applicable to any motion. Frame rate of a video is greater than 25 fps, and motion between two close frames are quite small; therefore, if the two images under estimation t0, t1 are two successive (or close) frames in the video sequence, the Approximate Algorithm is adequate enough for motion estimation. Part Ⅲ of this paper demonstrates experiments on both algorithms, and compares them with work in [17] (similar to [16]).

II. THEORY AND ALGORITHM

A. *Camera model: affine camera*

2D coordinates and 3D coordinates obey:

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = P \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} \qquad (1)$$

 ISBN: 978-1-61208-158-8

where $P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & 1 \end{bmatrix}$ is a matrix that determines a way of linear mapping from 3D space to 2D plane.

It should be noted that affine camera is the approximation of real cameras. It is applicable only when the depth changes of the interested object can be neglected compared to its depths.

B. Motion model of rigid-body

In 3D space, motion of rigid-body between two positions can be decomposed to rotation and translation. Rotation of rigid-body in 3D space can be described by a 3 x 3 matrix R, and translation can be described by a 3 x 1 matrix T.

$$T = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}, \quad R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

Let 3D coordinates of any feature point in image t0 be (X_1, Y_1, Z_1), and coordinates of this point in image t1 be (X_1', Y_1', Z_1'). The motion between t0 and t1 can be represented by:

$$\begin{bmatrix} X_1' \\ Y_1' \\ Z_1' \\ 1 \end{bmatrix} = M \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} \tag{2}$$

where the motion matrix is

$$M = \begin{bmatrix} R & T \\ O & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Note that an important property of rigid-body is, under a certain coordinate system, any point of the body has the same motion, i.e., the motion matrix. That is, every point pair obeys equation (2).

Although rotation matrix has 9 elements, it is an orthogonal matrix and subjects to 6 independent constraints:

$$\sum_{j=1}^{3} r_{ij}^2 = 1 (i = 1,2,3) \quad \sum_{k=1}^{3} r_{ik} r_{jk} = 0 (i, j = 1,2,3) \tag{4}$$

Therefore, there are only 3 independent parameters. Among various ways of expressing R by 3 independent parameters, the following 2 ways are commonly used:

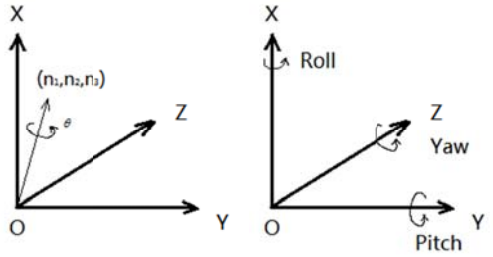
(i) Axis-angle representation

$$\begin{cases} r_{11} = \cos\theta + (1-\cos\theta) n_1^2 \\ r_{12} = (1-\cos\theta) n_1 n_2 - (\sin\theta) n_3 \\ r_{13} = (1-\cos\theta) n_1 n_3 + (\sin\theta) n_2 \\ r_{21} = (1-\cos\theta) n_1 n_2 + (\sin\theta) n_3 \\ r_{22} = \cos\theta + (1-\cos\theta) n_2^2 \\ r_{23} = (1-\cos\theta) n_2 n_3 - (\sin\theta) n_1 \\ r_{31} = (1-\cos\theta) n_3 n_1 - (\sin\theta) n_2 \\ r_{32} = (1-\cos\theta) n_3 n_2 + (\sin\theta) n_1 \\ r_{33} = \cos\theta + (1-\cos\theta) n_3^2 \end{cases} \tag{5}$$

The 3D vector from the origin $(0,0,0)$ to (n_1, n_2, n_3) represents the axis the rigid-body rotates around. The angle it rotates is θ.

(ii) Roll, Pitch and Yaw representation

$$R = \begin{bmatrix} \cos\theta_y \cos\theta_z & -\cos\theta_y \sin\theta_z & \sin\theta_y \\ \sin\theta_x \sin\theta_y \cos\theta_z + \cos\theta_x \sin\theta_z & -\sin\theta_x \sin\theta_y \sin\theta_z + \cos\theta_x \cos\theta_z & -\sin\theta_x \cos\theta_y \\ -\cos\theta_x \sin\theta_y \cos\theta_z + \sin\theta_x \sin\theta_z & \cos\theta_x \sin\theta_y \sin\theta_z + \sin\theta_x \cos\theta_z & \cos\theta_x \cos\theta_y \end{bmatrix} \tag{6}$$



(a) left: Angle-axis way of expressing 3D motion
(b) right: Roll-Pitch-Yaw way of expressing 3D motion
Figure 1. Two common way of expressing 3D motion

C. Solve the motion problem

Motion estimation can be completed in 2 steps.

First step: convert the problem of estimating motion to solving equation set.

From $\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = P \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix}$, $\begin{bmatrix} x_1' \\ y_1' \\ 1 \end{bmatrix} = P \begin{bmatrix} X_1' \\ Y_1' \\ Z_1' \\ 1 \end{bmatrix}$ and $\begin{bmatrix} X_1' \\ Y_1' \\ Z_1' \\ 1 \end{bmatrix} = M \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix}$

we have:

$$\begin{bmatrix} x_1' \\ y_1' \\ 1 \end{bmatrix} = PM \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} \tag{7}$$

Every point pair gives 2 equations:

$$\begin{cases} x_1' = p_{14} + p_{11}t_1 + p_{12}t_2 + p_{13}t_3 + X_1(p_{11}r_{11} + p_{12}r_{21} + p_{13}r_{31}) \\ \quad + Y_1(p_{11}r_{12} + p_{12}r_{22} + p_{13}r_{32}) + Z_1(p_{11}r_{13} + p_{12}r_{23} + p_{13}r_{33}) \\ y_1' = p_{24} + p_{21}t_1 + p_{22}t_2 + p_{23}t_3 + X_1(p_{21}r_{11} + p_{22}r_{21} + p_{23}r_{31}) \\ \quad + Y_1(p_{21}r_{12} + p_{22}r_{22} + p_{23}r_{32}) + Z_1(p_{21}r_{13} + p_{22}r_{23} + p_{23}r_{33}) \end{cases} \tag{8}$$

If motion is small enough for small-angle approximation, and if the Roll-Pitch-Yaw representation is adopted, the rotation matrix is simplified to:

$$R=\begin{bmatrix}1 & -n_3\theta & n_2\theta\\ n_3\theta & 1 & -n_1\theta\\ -n_2\theta & n_1\theta & 1\end{bmatrix}=\begin{bmatrix}1 & -\phi_3 & \phi_2\\ \phi_3 & 1 & -\phi_1\\ -\phi_2 & \phi_1 & 1\end{bmatrix} \quad (9)$$

where $(n_1\theta)^2+(n_2\theta)^2+(n_3\theta)^2=\theta^2$ .i.e. $\phi_1^{\,2}+\phi_2^{\,2}+\phi_3^{\,2}=\theta^2$ (10)

Rewrite the equation set (8):

$$\begin{bmatrix}x_1'-p_{14}-X_1p_{11}-Y_1p_{12}-Z_1p_{13}\\ y_1'-p_{24}-X_1p_{21}-Y_1p_{22}-Z_1p_{23}\end{bmatrix}=\begin{bmatrix}Y_1p_{13}-Z_1p_{12} & Z_1p_{11}-X_1p_{13} & X_1p_{12}-Y_1p_{11} & p_{11} & p_{12} & p_{13}\\ Y_1p_{23}-Z_1p_{22} & Z_1p_{21}-X_1p_{23} & X_1p_{22}-Y_1p_{21} & p_{21} & p_{22} & p_{23}\end{bmatrix}\begin{bmatrix}\phi_1\\ \phi_2\\ \phi_3\\ t_1\\ t_2\\ t_3\end{bmatrix}$$

Multiple point pairs can give equation set:

$$\begin{bmatrix}x_1'-p_{14}-X_1p_{11}-Y_1p_{12}-Z_1p_{13}\\ y_1'-p_{24}-X_1p_{21}-Y_1p_{22}-Z_1p_{23}\\ \vdots\\ x_n'-p_{14}-X_np_{11}-Y_np_{12}-Z_np_{13}\\ y_n'-p_{24}-X_np_{21}-Y_np_{22}-Z_np_{23}\end{bmatrix}=\begin{bmatrix}Y_1p_{13}-Z_1p_{12} & Z_1p_{11}-X_1p_{13} & X_1p_{12}-Y_1p_{11} & p_{11} & p_{12} & p_{13}\\ Y_1p_{23}-Z_1p_{22} & Z_1p_{21}-X_1p_{23} & X_1p_{22}-Y_1p_{21} & p_{21} & p_{22} & p_{23}\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ Y_np_{13}-Z_np_{12} & Z_np_{11}-X_np_{13} & X_np_{12}-Y_np_{11} & p_{11} & p_{12} & p_{13}\\ Y_np_{23}-Z_np_{22} & Z_np_{21}-X_np_{23} & X_np_{22}-Y_np_{21} & p_{21} & p_{22} & p_{23}\end{bmatrix}\begin{bmatrix}\phi_1\\ \phi_2\\ \phi_3\\ t_1\\ t_2\\ t_3\end{bmatrix} \quad (11)$$

The first matrix on the right side of the above equation is written down as A, which has a rank of no greater than 5. This is due to the limitation of single camera, because the camera cannot give information of depth along its optical axis. Below is the *proof*:

Observe matrix A. We can see the odd rows of the right three columns are identical to each other, and so are the even rows. Elementary row transformation operations are applied to A. Eliminate the right three columns of Row 2i-1 with the first row, and those of Row 2i with the second row（i=2,3…n）:

$$\begin{bmatrix}Y_1p_{13}-Z_1p_{12} & Z_1p_{11}-X_1p_{13} & X_1p_{12}-Y_1p_{11} & p_{11} & p_{12} & p_{13}\\ Y_1p_{23}-Z_1p_{22} & Z_1p_{21}-X_1p_{23} & X_1p_{22}-Y_1p_{21} & p_{21} & p_{22} & p_{23}\\ & \vdots & & 0 & 0 & 0\\ & \vdots & & & \vdots & \\ & \vdots & & 0 & 0 & 0\\ Y_np_{13}-Z_np_{12} & Z_np_{11}-X_np_{13} & X_np_{12}-Y_np_{11} & 0 & 0 & 0\\ Y_np_{23}-Z_np_{22} & Z_np_{21}-X_np_{23} & X_np_{22}-Y_np_{21} & 0 & 0 & 0\end{bmatrix} \quad (12)$$

Of the obtained matrix (12), the right three columns as a sub-matrix has a rank of 2 at most, because there are only 2 nonzero rows; the right three columns as a sub-matrix has a rank of 3 at most. So maxim rank of matrix (12) is 5. Elementary row transformation operations do not change the rank of matrix, so matrix A has a rank of 5 at most.

From the above we know motion estimation of rigid-body need at least 2 cameras. Let the projection matrix of the second camera be:

$$P=\begin{bmatrix}p_{11}' & p_{12}' & p_{13}' & p_{14}'\\ p_{21}' & p_{22}' & p_{23}' & p_{24}'\\ 0 & 0 & 0 & 1\end{bmatrix}$$

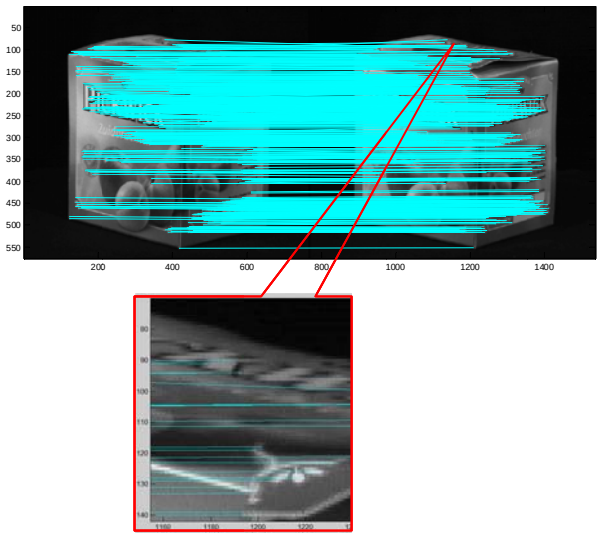
We have:

$$\begin{bmatrix}x_1'-p_{14}-X_1p_{11}-Y_1p_{12}-Z_1p_{13}\\ y_1'-p_{24}-X_1p_{21}-Y_1p_{22}-Z_1p_{23}\\ \vdots\\ x_n'-p_{14}-X_np_{11}-Y_np_{12}-Z_np_{13}'\\ y_n'-p_{24}-X_np_{21}-Y_np_{22}-Z_np_{23}'\end{bmatrix}=\begin{bmatrix}Y_1p_{13}-Z_1p_{12} & Z_1p_{11}-X_1p_{13} & X_1p_{12}-Y_1p_{11} & p_{11} & p_{12} & p_{13}\\ Y_1p_{23}-Z_1p_{22} & Z_1p_{21}-X_1p_{23} & X_1p_{22}-Y_1p_{21} & p_{21} & p_{22} & p_{23}\\ & \vdots & & & \vdots & \\ Y_np_{13}'-Z_np_{12}' & Z_np_{11}'-X_np_{13}' & X_np_{12}'-Y_np_{11}' & p_{11}' & p_{12}' & p_{13}'\\ Y_np_{23}'-Z_np_{22}' & Z_np_{21}'-X_np_{23}' & X_np_{22}'-Y_np_{21}' & p_{21}' & p_{22}' & p_{23}'\end{bmatrix}\begin{bmatrix}\phi_1\\ \phi_2\\ \phi_3\\ t_1\\ t_2\\ t_3\end{bmatrix}$$

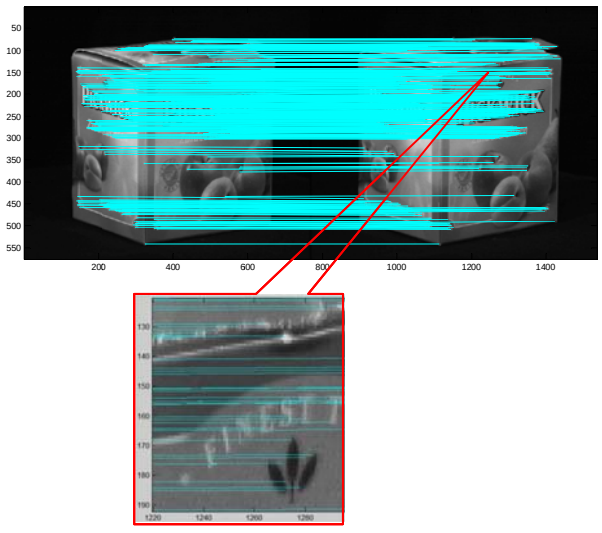
(13)

The matrix on the left is denoted as b . The matrix on the right is denoted as A, x successively. The rank of A can be 6 at most now. Thus, estimating motion is converted to solving equation set

$$b=Ax \quad (14)$$



(a) Matched point pairs in image t0 (left) and t1 (right) from camera 1



(b) Matched point pairs in image t0 (left) and t1 (right) from camera 2

Figure 2. Matched point pairs in image t0 and t1 from camera1 and 2

 ISBN: 978-1-61208-158-8

Use m point pairs from camera 1 (or view 1) and k point pairs from camera 2 (or view 2), on the condition that $n = m + k, (m, n \geq 1)$ ($m, n \geq 1$). In theory, 3 point pairs from two views are enough for solving the equation set. When n$\geq$3, we need to solve an equation set to derive the results. After 3D reconstruction with image t0 and image matching between image t0&t1, we can obtain a large number of feature point pairs. To exploit the information we have, we first use all point pairs to estimate motion. To enhance the algorithm's robustness, use the initial motion matrix for reprojection, compare the results with real 2D coordinates in image t1, and eliminate outliers whose errors are larger than threshold 10 (experiment shows error is normally less than 5, if error >10 appears, it is probably an outlier). Appearance of outliers may be the result of wrong matching.

Second step : solve the equation set by least mean square error principle.

To solve the equation set by least mean square error principle, QR decomposition method is adopted in this paper.

Because of the existence of error, $Ax = b + \varepsilon$. The problem equates solving x that minimizes norm $\|\varepsilon\|_2^2$. We can find Q that obeys $QA = \begin{pmatrix} R \\ O \end{pmatrix}$, where Q is an orthogonal matrix and R is a nonsingular upper triangular matrix.

$$\|Ax-b\|_2^2 = \|Q(Ax-b)\|_2^2 = \|QAx-Qb\|_2^2 = \left\| \begin{bmatrix} R \\ O \end{bmatrix} - Qb \right\|_2^2$$

Write $Qb = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$, then we have $\|Ax-b\|_2^2 = \left\| \begin{bmatrix} Rx-b_1 \\ b_2 \end{bmatrix} \right\|_2^2$. It is a column vector, and thus $= \|Rx-b_1\|_2^2 + \|b_2\|_2^2$. Notice that $\|b_2\|_2^2$ is constant; therefore, the original problem of minimizing $\|Ax-b\|_2^2$ converts to minimizing $\|Rx-b_1\|_2^2$, namely $Rx - b_1$ =0.

Thus, we have obtained x vector. For equation set (13), x= $(\phi_1, \phi_2, \phi_3, t_1, t_2, t_3)^T$

D. Universal Algorithm

If there is no small-angle approximation, we can write equation set as follows:

$$\begin{bmatrix} x_1'-p_{14} \\ y_1'-p_{24} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_n'-p_{14}' \\ y_n'-p_{24}' \end{bmatrix} = \begin{bmatrix} X_1p_{11} & X_1p_{12} & X_1p_{13} & Y_1p_{11} & Y_1p_{12} & Y_1p_{13} & Z_1p_{11} & Z_1p_{12} & Z_1p_{13} & p_{11} & p_{12} & p_{13} \\ X_1p_{21} & X_1p_{22} & X_1p_{23} & Y_1p_{21} & Y_1p_{22} & Y_1p_{23} & Z_1p_{21} & Z_1p_{22} & Z_1p_{23} & p_{21} & p_{22} & p_{23} \\ & & & & \cdot & & & & & & & \\ & & & & \cdot & & & & & & & \\ & & & & \cdot & & & & & & & \\ & & & & \cdot & & & & & & & \\ & & & & \cdot & & & & & & & \\ & & & & \cdot & & & & & & & \\ & & & & \cdot & & & & & & & \\ X_np_{11}' & X_np_{12}' & X_np_{13}' & Y_np_{11}' & Y_np_{12}' & Y_np_{13}' & Z_np_{11}' & Z_np_{12}' & Z_np_{13}' & p_{11}' & p_{12}' & p_{13}' \\ X_np_{21}' & X_np_{22}' & X_np_{23}' & Y_np_{21}' & Y_np_{22}' & Y_np_{23}' & Z_np_{21}' & Z_np_{22}' & Z_np_{23}' & p_{21}' & p_{22}' & p_{23}' \end{bmatrix} \begin{bmatrix} r_{11} \\ r_{21} \\ r_{31} \\ r_{12} \\ r_{22} \\ r_{32} \\ r_{13} \\ r_{23} \\ r_{33} \\ t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad (15)$$

Use m point pairs from camera 1 (or view 1) and k point pairs from camera 2 (or view 2), on the condition that $n = m + k, (m, n \geq 1)$. The method of solving equation set is the same as Approximate Algorithm. For equation set (15), $x = (r_{11}, r_{21}, r_{31}, r_{12}, r_{22}, r_{32}, r_{13}, r_{23}, r_{33}, t_1, t_2, t_3)^T$.

For Universal Algorithm, to ensure A has full rank, at least 4+4=8 pairs of feature points are needed.

III. EXPERIMENTAL VALIDATION

Validity of the proposed algorithm was verified by MATLAB experiments. We used images caught by 2 cameras.

Below are the steps of the algorithm.

---

1. {**3D Reconstruction**}
2. {**Match image t0 and t1**. Obtain 2D coordinates of feature points in t1.}
3. {**Binocular Vision**: 3D reconstruction and image matching for 2 cameras}
4. {**Initialization**: input m pairs of camera1 and k pairs of camera2)}
5. {**Turn motion estimation into solving equation set.** Every point pair gives 2 equations. }
6. {**Solve equation set** $b = Ax$ under LSQ criterion (by QR decomposition).}
7. {**Reproject** with derived M, compare 2D results to real 2D coordinates.}
   If (Errors of all points are under threshold) then
   {Eliminate point pairs whose errors exceeds threshold.
   Go to 5, repeat 5, 6, and 7}
   Else {Give final results}
8. {**Live Display of 3D Object}**

A. 5° around axis x (both algorithms)

A box was known to have rotated an angle of 5° . For simplicity, the motion in our experiment only had rotation component. That is, t1, t2 and t3 in the translation matrix should be all zeros in theory, and the emphasis of result should be the rotation matrix. First, the Approximate Algorithm was tested. Second, the Universal Algorithm, which is rigorous, was tested. According to (6), $\theta_x = \arctan(-r_{23} / r_{33})$.

We adopted two images shown in Figure 2. Reconstruct image t0 in 3D space, and match image t0 and t1, we obtained 318 point pairs from camera 1in all. Similarly, we obtained 246 point pairs from camera 2 in all.

Experiments with different input numbers of point pair n were conducted. Input point pairs were results of equal-interval sampling from all point pairs. The calculated t1, t2, t3 were all approximately zeros; calculated θ_x is demonstrated in Figure 3.

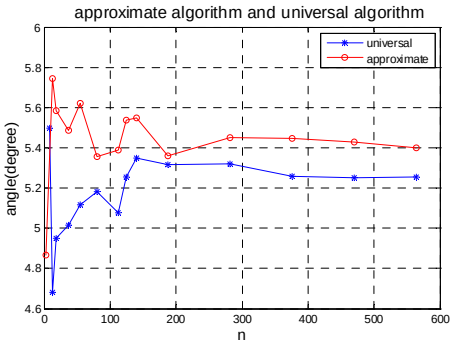


Figure 3. Experiments with different number of input point pairs

As shown, when input point pairs are barely enough for calculation, results are greatly influenced by the choice of input data, and are unreliable; however, when there are enough input data, both the Approximate Algorithm and the Universal Algorithm can give stable results (5.4 and 5.3). When input number of point pairs is greater than 200 (two cameras), for the Approximate Algorithm, fluctuations of results do not exceed 0.05 / 5.4=0.9%, and for the Universal Algorithm, fluctuations are within 0.06 / 5.2559= 1.2%.

It was also verified that reprojection errors were almost all less than 2 pixels along any axis in the image for 318+246=564 point pairs. Figure 4 shows an example of reprojection errors along x axis. In addition, the algorithm can eliminate outliers, whose reprojection errors are greater than threshold 10, thus guaranteeing stable and robust results.

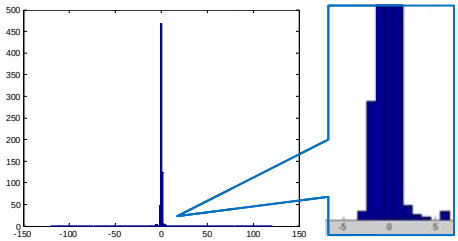
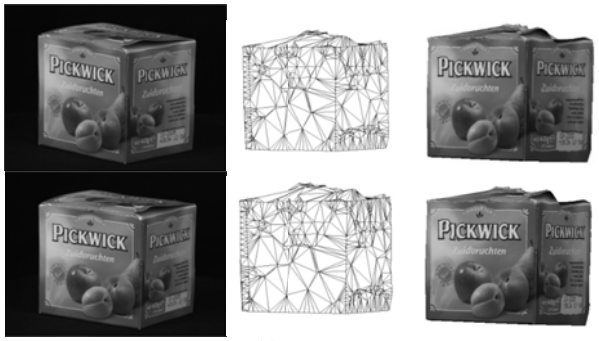


Figure 4. reprojection errors distribution

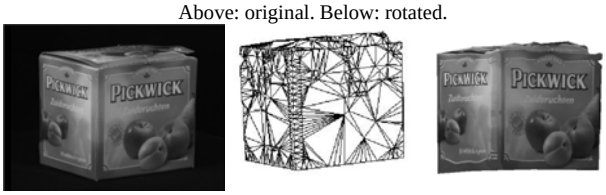
B. *10° around axis x (Universal Algorithm only)*

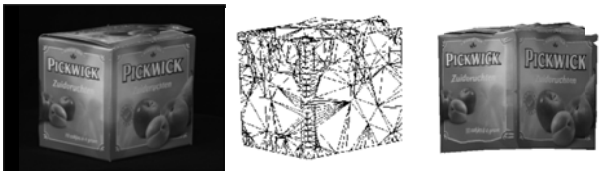
The box was known to have rotated about 10° around axis x in Figure 1. Similar to experiment 1, there was merely rotation component of the motion. Only the Universal Algorithm can be applied. 210 point pairs from camera 1 and 168 pairs from camera 2 were used. The calculated angle was 10.6700.

C. *Live display of 3D object*



(a) camera 1
Above: original. Below: rotated.





(b) camera 2
Above: original. Below: rotated.

Figure 5. comparison of real object and displayed object

Our process of live display of 3D object is as follows: (i) Add the calculated motion to the original 3D object which has been reconstructed, i.e. apply equation (2) to all points on the object. (ii) Connect all points in meshes to obtain the surface. (iii) Attach the texture of the original object to the meshed surface.

Figure 5 shows 10° rotation of real object and displayed object from 2 views in experiment B. (a) presents the pictures from camera 1 and, and (b) presents those from camera 2. The left column shows real pictures shoot by corresponding cameras. The middle column shows 3D object grids before attaching texture. The right column shows displayed object with texture attached.

As shown, from both views, the displayed object accords with its prototype in real world.

3D object can be displayed in free-angle as long as the original object is entirely reconstructed. We can reconstruct the object using 3 frames of a video sequence, and display any frame within certain range of the video sequence. The range is decided by the maximum angle of image matching.

D. *Comparison with recent work and merits discussion*

To further study accuracy of the proposed method, control experiments were conducted from many different viewpoints. We adopted Han-Kanade method [17] to reconstruct a stereo-object as well as recovering motion, and then adopted the proposed method to calculate motion using same input points. As shown in Figure 6, our method is comparable to Han-Kanade method in terms of precision.

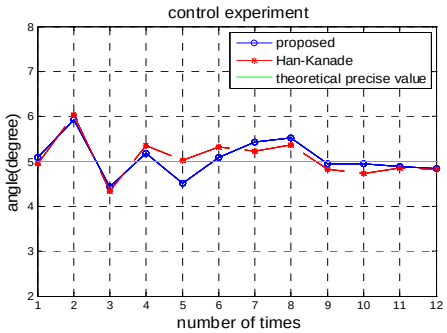


Figure 6. comparison with Han-Kanade method (same input point pairs)

Thorough recovery of 3D structure and motion from uncalibrated cameras is quite time-consuming. On a personal computer of 2.4GHz master frequency, it cost 5~10 minutes for one time of 3-frame reconstruction. Stereo reconstruction from 5 views will take much more time. Moreover, we cannot reduce the number of cameras, since fewer cameras will bring

 ISBN: 978-1-61208-158-8

more difficulties on image matching because of wider baseline between every two camera. This directly affects 3D reconstruction. The proposed method, on the other hand, costs less than one second with same number of input points. (Same to Han-Kanade method, this does not include the time for display, making it a fair comparison) Evidently, it would save a great deal of time in case of long video sequence if we recover motion using proposed method, rather than recover structure and motion simultaneously over time.

E. Simple error analysis

Errors include: (1) Quantization error. Quantization error exists throughout the whole process of computer vision including motion calculation problem in this paper. (2) 3D reconstruction errors, including multi-view correspondence error, etc. As input of motion vector calculation, 3D reconstruction with errors will lead to errors in the motion vector. (3) Error in matching image t0 and t1. Finite pixel resolution may introduce error, and matching algorithm could also influence accuracy in 2D coordinate of image t1.

Additionally, camera calibration and computational accuracy have little effects on results of our method. In our method, 3D reconstruction is based on uncalibrated cameras, and motion calculation does not involve calibration either. Computational accuracy of MATLAB is more than 40 decimal places; therefore, truncation error has no influence on our results.

IV. CONCLUSION AND FUTURE WORK

Several conclusions can be drawn from the experiment results.

Firstly, the correctness and feasibility of the proposed algorithm are verified. For the Approximate Algorithm, exactly 3 point pairs are needed for motion estimation. Of course, the result depends largely on the selected 3 point pairs, and is too sensitive and unstable.

Secondly, the algorithm is stable and robust when there is enough input data, as results are obtained on a unanimous ground. The algorithm can eliminate outliers to a certain extent.

Thirdly, since the algorithm merely calculates motion vector but not simultaneously recover motion and structure, it is quite succinct. Time cost depends on the number of point pairs. For a typical experiment with about 200 point pairs each camera, both the Universal Algorithm and the Approximate Algorithm cost less than a few second. Thus, static 3D reconstruction can be performed only at intervals, while motion vector is calculated and added to the static model at any time during the intervals. Compared to conducting 3D reconstruction over time, this scheme will save much time.

Future work will concentrate on the following several aspects:

Selecting a proper interval between every two times of motion vector calculation: strike the balance between time consumption and necessity;

Improving the continuity of real-time display, perhaps by interpolating;

Testing cases when motion calculation is limited by wide-baseline image matching， in order to find out the limitation of application.

V. ACKNOWLEDGEMENT

This paper is supported by the Fundamental Research Funds for the Central Universities, Number: 1107021051, and National Natural Science Funds of Jiangsu Province of China, Number: BK2010386.

REFERENCES

[1] T. Papadimitriou, M. G. Strintzis, and M. Roumeliotis, “Robust Estimation of Rigid Body 3-D Motion Parameters Based on Point Correspondences”, IEEE Trans. on Circuits And Systems for Video Technology, 2000, pp. 541-549.

[2] J. Weng, T. S. Huang, N. Ahuja, “Motion and Structure From Two Perspective Views: Algorithms, Error Analysis, and Error Estimation”, IEEE Trans. on Pattern Analysis and Machine Intelligence, 1989, pp. 451-476.

[3] R. Hartley, “In Defense of the 8-point Algorithm”, Proc. 5th ICCV, Cambridge, USA, 1995, pp. 1064-1070.

[4] H. Ebrahimnezhad, H. G. Robust, “Motion From Space Curves and 3D Reconstruction From Multiviews Using Perpendicular Double Stereo Rigs”, Image and Vision Computing, 2008, pp. 1397–1420.

[5] J. M. M. Montiel, J.D. TardoHs, and L. Montano, “Structure and Motion From Straight Line Segments”, Patten Recognition, 2000, pp. 1295-1307.

[6] E. G. Steinbach, P.Eisert, and B. Girod, “Model-based 3D Shape and Motion Estimation Using Sliding Textures”, Proc. Vision Modeling and Visualization Conference, 2001, pp. 375-382.

[7] J. Neumann, C. Fermuller, and Y. Aloimonos, “Polydioptric camera design and 3D motion estimation”, Proc. IEEE Conf. Computer Vision and Pattern Recognition, volume 2, 2003, pp. 294-301.

[8] J. W. Roach and J. K. Aggarwal, “Determining the Movement of Objects From A Sequence of Images”, IEEE Trans. on Pattern Anal. Machine Intell, 1980, pp. 554–562.

[9] J. Oliensisa, “Multi Frame Structure from Motion Algorithm under Perspective Projection”, Int J. Comput Vis, 1999, pp. 163–192.

[10] P. Sand and S. Teller, “Particle Video: Long-Range Motion Estimation Using Point Trajectories”, Int J. Comput. Vis, 2008, pp. 72–91.

[11] B. Matei and P. Meer, “A General Method for Errors-in-Variables Problems in Computer Vision,” Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2000, pp. 18–25.

[12] P. Firoozfam and S. Negahdaripour, “Theoretical Accuracy Analysis of N-Ocular Vision Systems for Scene reconstruction, Motion Estimation, and Positioning Problems in Computer Vision”, Proc. 2nd Int. Symposium on 3D Data Processing, Visualization, and Transmission, 2004, pp. 888-895.

[13] Y. Sheikh, A. Hakeem, and M. Shah, “On the Direct Estimation of The Fundamental Matrix”. Proc. IEEE Conf. Comput Vis. Patt. Rec., 2007, pp. 1–7.

[14] P. Chen, “Why Not Use the Levenberg–Marquardt Method For Fundamental Matrix Estimation”, IET Comp. Vis., Vol. 4, Iss. 4, 2010, pp. 286–294.

[15] H. P. Wu and S. H. Chang, “Fundamental Matrix of Planar Catadioptric Stereo Systems”, IET Comput. Vis., Vol. 4, Iss. 2, 2010, pp. 85–104.

[16] R. Szeliski, “Computer Vision: Algorithms and Applications”, Chapter 7, 2010, pp. 343-374.

[17] M. Han and T. Kanade, “Creating 3D Models with Uncalibrated Cameras”, Proc. IEEE Computer Society Workshop on the Application of Computer Vision, 2000, pp. 178-185.

 ISBN: 978-1-61208-158-8

Patterns of Emotion Driven by Affect State and Environment

An architecture for a visualized, independent, autonomous, learning agent

Paul G. Joseph and Haim Levkowitz
Computer Science Department
University of Massachusetts
Lowell, MA., USA
pjoseph@gmail.com, haim@cs.uml.edu

***Abstract*—The neuroscientist Jaak Panksepp posits the existence of seven physical systems in the mammalian brain, that when simulated in the laboratory, result in emotions identical to known emotions. These seven systems are SEEKING, RAGE, PANIC, FEAR, LUST, CARE, and PLAY. From the perspective of mathematics, these emotions form the dimensions of a phase space in which every emotion is located and modeled as a dynamical system. Inputs from outside the system interact with internal values and inform the dynamical system, resulting in a simulated affect that in turn drives observed patterns of emotion. This paper discusses a framework built to explore this premise. As a first cut it uses a linear seven-variable dynamical system to drive a search heuristic for a utility-based reflex agent. The results indicate that even this simple linear model shows the basic patterns of emotion observed in mammals. These patterns of emotion facilitate discovery and decision. It provides an independent "always on, automatic" discovery and decision system capable of adaptive goal setting, which works without the need for significant additional cognitive analysis. It offers the basis of a framework that is extendable to include additional sensory information, learning, memory/persistence, and an independent cognitive system.**

Keywords - patterns; affect; emotions; Panksepp; autonomous agent; mammalian emotion.

I. INTRODUCTION

Rosalind Picard [30] used the terms affective computing systems and emotion-oriented computing to refer to systems that consider emotions. Emotion handling is fundamentally important, and a significant amount of research in this area has been carried out in connection with real-life non-simulated pervasive computer systems [4][5][12][23][27][28][33][38]. To date, in the field of software in general, the bulk of the work in this area has been to examine patterns of emotion (facial expressions, bodily movements, language) to infer internal emotions.

In this paper, however, we seek to lay out the basis for a complementary approach—one that models neural systems posited to exist in mammalian brains in order to generate and manage affect. We use dynamical system theory to model these systems and to generate internal affect states that together resulting in an overall "emotion" that in turn drives behavior.

The premise of this paper is that "meaning" substantially involves an emotional evaluation of an object based on our interaction with the object. We use affect values generated by a dynamical system as the basis of a heuristic used by a utility agent to navigate its environment. The premise is that with this approach, it may be possible to attach affect to an object and, thereby, give a computer-based agent a sense of meaning of the objects it is working with.

This, in turn, allows the agent the ability to perform adaptive goal setting. Depending on circumstances and its current needs, the agent will demonstrate appropriate patterns of emotion and act based on these patterns, e.g., if FEAR is high, it will run away from the object causing the fear. If, however, SEEK predominates, then it will seek and if hungry it will specifically seek for nourishment. In short, goals are adaptive and based on the current emotional state.

Though this research explores a model of the mammalian affective system, its goal is not to explore emotions per se. In addition, though it uses the framework of a game to explore the model, the purpose of this work is neither to develop better gaming theory nor to develop a more robust algorithm for the game. Rather, the purpose of this research is to see if, using the model, a computer-based agent can develop affect with respect to other entities it encounters based on interactions with them, and based on this affect, generate appropriate emotion patterns that produce adaptive goal setting, discovery, and decision.

II. BASIS OF APPROACH

The basis of the approach is to model findings from neuroscience using dynamical systems and to use the resulting model as the search heuristic for a utility agent that allows the agent to develop affect for objects in its environment. Subsequently, this affect generates emotional patterns that result in adaptive goal setting and behavior.

 ISBN: 978-1-61208-158-8

A. *Neuroscience: Empirically Based Fundamental Emotions*

Dr. Jaak Panksepp, a neuroscientist from Washington State University proposed the existence of seven core emotional systems, because they "generate well-organized emotion sequences that can be evoked by localized electrical stimulation of the brain" [29]. We provide a very brief sketch of the Panksepp model (which draws directly from Panksepp, 1998):

SEEKING system: a network that promtes survival activities, making creatures intensely interested in exploring their world, and to get excited when they get what they desire. Neuroanatomically they correspond to the major self-simulation system that courses from the midbrain up to the cortex.

RAGE system: works in opposition to SEEKING, and mediates anger. It is aroused by frustration and attempts to curtail freedom of action. It parallels the trajectory of the FEAR system.

FEAR system: probably designed during evolution to help reduce pain and possibility of destruction. When stimulated intensely, it leads creatures to run away; when stimulated weakly, it causes the creature to freeze.

PANIC system: governs social attachment emotion—specifically related to absence of maternal care when the creature is a baby.

Additionally, there are three "special-purpose" emotions. Panksepp calls these emotions "more sophisticated, special purpose, socioemotional systems that are engaged at appropriate times in the life of all mammals." Currently far less understood than the other four, these are:

LUST system: involves sex and sexual desire, evolved for species propagation

CARE system: maternal love and caring

PLAY system: produces the "roughhousing play of all young animals and humans" at some stage in their development to facilitate learning.

There is by no means universal agreement or acceptance of Panksepp's model. Rebuttals and alternate formulations [1][2][6][7][8] have been proposed. Chiefly these differ in the number and kind of basic emotions. This paper does not seek to prove the correctness of Panksepp's particular formulation. Rather it leverages the fact that one can implement any such model compactly and usefully as a dynamical system.

A key feature of Panksepp's model is its empirical basis for "fundamental" emotions, which distinguishes it from numerous other systems that also claim to have identified so-called "fundamental" emotions. If the physical circuitry for any given emotion does not exist in the brain, then this emotion is a combination of those emotions that do have physical representation.

Thus, for example, "romantic love" would be viewed as some combination of (perhaps) CARE, PLAY, SEEK, and LUST; resentment or indignation as combinations of RAGE, CARE, SEEK. The SEEK response, associated with a large release of dopamine and a sense of well-being would cause emotions like "joy", "satisfaction", "contentment", "enjoyment", etc., variations of the theme of gratification of the SEEKING system and the attainment and material benefit of the goal being sought.

Greene [18][19] describes his theory that the human brain has two modes of responding to situations. One, a set of efficient automatic responses driven by emotions, the other a manual mode used in response to non-standard situations that involves significant cognitive evaluation. LeDeux [25] posits that the former typically operates in timeframes of about 10 ms, while the latter in timeframes closer to 500ms. The non-cognitive bias in the Panksepp based dynamical system search heuristic proposed above allows for an "automatic, always on and very quick" response corresponding to that described by both Greene and LeDoux.

B. *Mathematics: Dynamical Systems*

At the heart of the framework is the dynamical system model. See [37], for example, for additional details on dynamical systems. Dynamical systems have modeled a wide range of systems including marriage [17] and emotional development [3]. In particular, Lewis [26] used it to bridge neurobiology and emotion theory while Scherer, [35] modeled emotions as emergent processes. However, neither used these approaches for software agents, nor used a model such as the Panksepp model, whereas here, the dynamical system explicitly models in a software agent, the Panksepp systems namely: SEEKING, RAGE, FEAR, PANIC, LUST, CARE, and PLAY.

As per Panksepp's description of these basic emotions, agents when hungry will SEEK food; depending on age, they may PLAY or if they are parents, may CARE or LUST if they are young adults, and do all this using SEEK. Eventually LUST could result in progeny. If a hostile agent is in the vicinity, then the agent experiences FEAR. If trapped, it experiences RAGE. If a child is at a distance from its parent or its parent is not available then it experiences PANIC.

An agent's current emotional state is its existing state modified by incoming changes in affect caused by the meaning of the current objects in its vicinity (defined by how they affect the agent's current emotional state).

As mentioned earlier, Panksepp's model is qualified, and is currently the subject of considerable debate in the affective neuroscience community. However, this paper takes the pragmatic approach that it does not really matter if eventually it is found that there are four mechanisms or ten mechanisms—the important finding is that there is a relatively small number of them and that they have both a physical basis and a readily understood meaning.

We distinguish this approach from other AI approaches, such as neural nets where the meaning of the nodes is not well defined or known. This clear definition of the variables and their meaning in turn allows for an intuitive understanding of system emotion. In addition, we distinguish the model from "Braitenberg architectures" [24] in that here, the basis is an actual, specific, neuroscience model—that of the mammalian brain.

 ISBN: 978-1-61208-158-8

III. IMPLEMENTATION

The Pacman game [9] offers a convenient framework for a preliminary exploration of the approach suggested in this paper. Pacman used for its search heuristic, the systems identified by Panksepp, modeled as a dynamical system. The heuristic is coded to make Pacman a utility reflex agent [34] aiming to maximize its overall well-being with actions based on the current location of the ghosts and food. We initialized Pacman's dynamical system with random values between a minimum of 0 and a maximum of 100. Additionally, Pacman had other properties, such as gender, health, age, and strength fixed at "birth," i.e., initialization.

As a first cut, certain relationships are hard-coded into the model. This hard coding is analogous to behavior Pacman was "born" with, i.e., a result of evolution. For example, the model hard coded Pacman's SEEKING emotion as inversely proportional to distance to food, and FEAR to be inversely proportional to the distance to the predator (ghost). In this first cut, we arbitrarily chose the proportionality values to be 1/1000.

Later models will explore if Pacman can learn these distance relationships instead of simply being "born" with them., the learning being assisted by a combination of persistence of emotions with respect to encountered objects and information shared globally across all instances of Pacman.

As a first cut, Pacman used the following simple linear, additive, dynamical system model, with each emotion being described as some linear combination of existing emotions:

```
pacman.SEEKING += 1000/foodDistance + 1000/pacman.AGE + pacman.HUNGER + pacman.RAGE + pacman.PANIC + pacman.FEAR + pacman.PLAY + pacman.LUST
pacman.FEAR += 1000/ghostDistance + health + 1000/strength + pacman.RAGE + pacman.PANIC
pacman.RAGE += 1000/ghostDistance + 1000/health + 1000/strength + pacman.FEAR + pacman.RAGE + pacman.PANIC
pacman.PANIC += 1/ghostDistance + 1000/health + 1000/strength + pacman.FEAR + pacman.RAGE
pacman.CARE += 1000/foodDistance + 1000/ pacman.RAGE + 1000/pacman.FEAR + pacman.PLAY + pacman.AGE + health + gender
pacman.PLAY += 1000/foodDistance + 1000/pacman.RAGE + 1000/pacman.FEAR + 1000/pacman.AGE + health
pacman.LUST += 1000/foodDistance + 1000/pacman.RAGE + 1000/pacman.FEAR + pacman.PLAY + 1000/pacman.AGE + health
```

The emotional state of Pacman was then calculated as the simple summation of the "positive" dimensions SEEKING, CARE, PLAY, and LUST , less the "negative" dimensions of FEAR, PANIC, and RAGE. i.e..:

```
pacman.EMOTION = (pacman.SEEKING + pacman.CARE + pacman.PLAY + pacman.LUST) - (pacman.FEAR + pacman.PANIC + pacman.RAGE)
```

A simple loop recalculated the model's equations continuously. Figure 1 shows a typical screenshot from the running game. Figure 2 shows values for each of the seven emotions from a sample run of Pacman—the X-axis is the iteration number, the Y-axis the affect value.

The ghost in the game that hunts Pacman used a random algorithm to determine the direction of its next step. Because of this, while for any given iteration number, the values of the dynamical system across hundreds of runs are different, the patterns of emotion driven by the affect states (shown for one run in Figure 2) is the same for all runs.

IV. RESULTS

The single bands shown for the emotion patterns driven by SEEKING, LUST and PLAY are expected; Pacman is generally SEEKING and has no companions to PLAY with or LUST after! Likewise, the dual banding of emotion patterns driven by FEAR is the result of high FEAR when the ghosts were nearby and a switch to low FEAR when the ghosts were beyond a certain "threshold distance." The pattern of emotion for CARE varied inversely with that of FEAR and so showed a similar dual banding. The triple banding of emotion driven by PANIC and RAGE reflects an average value with fluctuations to either side when the ghosts were closer or further than some threshold value.

The last graph in Figure 2 reflects Pacman's fluctuating emotion, driven by his likewise fluctuating emotional state as it tried to stave off death.

It became clear that given several of the physical factors with which Pacman was initialized (age, fear, health, gender etc.), that even this simple linear model offered nearly infinite possibilities, implying that mathematically, one may not need to use anything more complex than a linear dynamical systems model in order to see rich behavior.

The non-cognitive bias of the model results in a very lightweight decision engine analogous to findings described by LeDoux [25] that emotional responses are at least an order of magnitude quicker than conscious, cognitive evaluations. Additionally, by definition, dynamical system models appear to closely correspond to key concepts from philosophy of mind including: total net affect state (emotion), being some combination of various magnitudes of basic affect states, it is "ineffable," a requirement specified for qualia [10]; the principle of marginal control where a higher level (emotion) is both defined by and controls a lower level (the constitutive emotions) [31]; correspondence between the model's ability to change values while at the same time have a definitive actionable value, and William James's concept of the transitive and substantive [22].

V. DISCUSSION

The thrust of this paper has been to use a model that leverages current findings from neuroscience on how the human brain develops affect with respect to objects or other living entities in its vicinity. The premise is that if this can be replicated in a computer or embodied in a robot, then the computer or robot can develop affect for an object and thereby a sense of meaning of the object. Additionally, expected to hold true is the reverse—previously understood sense of meaning for an object in turn regenerate affect values previously associated with the object based on

 ISBN: 978-1-61208-158-8

experience and the relative immediacy of the current interaction.

The drawback in the initial version of the equations used above, a "catch 22" as it were, is the hard coding of the distance variables. This considerably weakens the argument as possibly, without emotions, Pacman could do a better job of avoiding the ghost than with emotions. However, the point of the paper is not whether Pacman will do a better job with a "pure distance" based heuristic, but rather whether Pacman can be made to develop "affect" for the ghost, i.e., whether, at the end, Pacman has obtained a "meaning" of the ghost. Future work will aim at removing the hard coding of these distance variables and to see if through a combination of memory/persistence, using multiple instances of Pacman and the ghost, and observing ghosts consuming other instances of itself, whether Pacman can evolve similar distance relationships.

Another question is whether having values on affect dimensions for different objects in the vicinity suffice to assign true affect to that object, or does affect come into play only when there is a "body" to feel. Additionally, if needed, must a body be made of organic material such as flesh and blood or would a machine with hydraulics, sensors, and other artificial materials suffice.

Numerous researchers have raised questions related to embodiment, specifically, the symbol-grounding problem [20][36], the frame problem, the common-sense problem [21], and the rule-described/expertise problem [13]. It is the subject of much research (see for example [11][14][15][16]).

This paper takes the pragmatic approach that embodiment helps "seed" the system with meaningful attributes to real objects and that subsequently, this embodiment provides "agency" to the system; that mammals are, among other things, "organic implementations of algorithms" with senses acting as analog counters of values monitored to evaluate Panksepp well-being. The reason for the various differences in senses (counters) is merely to distinguish between different input data types. This implies that digital implementations of environment sensors would be analogous to organic implementations of the mammalian senses, or that both philosophically and practically, robots may have sufficient embodiment to support an equivalent of emotions. However, robots presently lack the dexterity needed to interact with an environment in a rich manner and consequently our approach is to simulate embodiment in a virtual environment using visualized agents.

Having few, specific and known, labels for the key variables allows for a physical feel for and an intuitive understanding of emotion, and contrasts against approaches such as neural networks. However, the question remains—how do we know it "works?" No clear answer is presently available—only the general and vague criterion that emotion seems "realistic." Popper [32] defined a hypothesis as scientific only if it is falsifiable. At present, we do not meet this criterion, i.e., the subject at hand does not presently qualify as "science."

Another (relatively minor) problem is related to the richness of the environment—it was thought that having only Pacmen, ghosts and food would not be sufficient and that additional objects would be needed in order to elicit rich emotions from the agents. However, compared to the earlier objections, this issue is of relatively secondary importance. By giving ghosts their own heuristic, by giving agents "memory" of objects they have interacted with together with emotional values attributed to these objects and the ability to store/persist them and share this knowledge with others of their "species", by allowing agents to "breed", have varying "strengths" etc., a large amount of richness can be introduced as needed. The hope is that by leveraging these additional (yet unused) options, Pacman will automatically evolve behaviors presently hard coded in the equations.

Lastly, studies by Greene [18][19] suggest a dual process theory in the human brain with one set of brain structures dealing with affect based decisions and another with utilitarian reasoning. In the case of strong competition/conflict between the two, this conflict is resolved in the ventromedial prefrontal cortex, resulting in a unified decision. There is no reason why a robot or agent cannot have a similar mechanism—a parallel and independent, analytical engine for utilitarian reasoning, with the final decisions reached by weighing recommendations based on affect against those based on utility.

VI. CONCLUSION

A simple, linear, lightweight, "automatic/always on" dynamical systems model, capable of adaptive goal setting, discovery, and decision. and resting on a combination of basics from dynamical systems, computer science, and neuroscience, appears to demonstrate seemingly reasonable patterns of emotion.

Four key concepts underlie the approach. First, to prevent ad hoc decision models, the approach attempts to follow closely theories of the mammalian brain. Second, the model has a strong non-cognitive bias. Third is the use of a dynamical systems model, which by definition, meets key requirements from philosophy of mind. Fourth, is the attempt to assign entities interaction dependant affect, as a way to approach the meaning/qualia problem.

Future work will: Evolve the distance relationships; Refine the model (specifically, further reduce arbitrariness and determine the parameters/coefficents) using additional findings from neuroscience; Add learning using information sharing and persistence; Explore use in a visualized agent as a lightweight decision system able to adaptively set goals and with a parallel cognitive system.

ACKNOWLEDGMENT

The authors thank Professors F. Martin and H. Yanco of the Computer Science Department, Professor W. Kaufmann of the Philosophy Department, and Professor J. Graham-Eagle of the Mathematics Department, all at the University of Massachusetts, Lowell, MA, for support, detailed reviews, and feedback. Thanks also to Mr. S. Cronin and Mr. R. Cole for valuable comments.

REFERENCES

[1] Barrett, L. 2006. Are emotions natural kinds? Perspectives on Psychological Science 1, 28-58.

[2] Barrett, L., Mesquita, B., Ochsner, K. N., and Gross, J. J. 2007. The Experience of Emotion. Annual Review of Psychology, Volume 58, 373-403.

[3] Camras, L. A. and Witherington, D. C. 2005. Dynamical systems approaches to emotional development. Developmental Review, 25(3-4), 328-350.

[4] Camras, L. A. and Shutter, J. 2010. Emotional Facial Expressions in Infancy. Emotion Review Vol. 2 no. 2, 120-129.

[5] Chen, C. Y., Forlizzi, J., and Jennings, P. (2006) ComSlippter: An Expressive Design to Support Awareness and Availability, Alt.CHI Paper in Extended Abstracts of Computer Human Interaction (ACM CHI 2006).

[6] Damasio, A. 1994. Descartes Error: Emotion, Reason, and the Human Brain, New York, New York: Putnam Publishing.

[7] Damasio, A. 1999, The Feeling of What Happens: Body and Emotion in the Making of Consciousness, New York, New York: Harcourt Brace and Company.

[8] Damasio, A. 2003, Looking for Spinoza: Joy, Sorrow, and the Feeling Brain, Boston, Mass: Houghton Mifflin Harcourt, Inc.

[9] DeNero, J. and Klein, D. 2010. Teaching Introductory Artificial Intelligence with Pacman. Symposium on Educational Advances in Artificial Intelligence.

[10] Dennett, D. 1988. Quinning Qualia. In Consciousness in Modern Science, A. Marcel and E. Bisiach, eds, Oxford: Oxford University Press.

[11] deVega, M., Glenberg, A., and Graesser, A. eds. 2008, Symbols,Embodiment, and Meaning, Oxford: Oxford University Press.

[12] Doulamis, N. 2006. An Adaptable Emotionally Rich Pervasive Computing System, Procedings of the Eusipco Conference.

[13] Dreyfus, H. 1992, What Computers Still Can't Do: A Critique of Artificial Reason, Cambridge, Mass: MIT Press.

[14] Franklin, S. 1997a. Autonomous Agents as Embodied AI Cybernetics and Systems. Special issue on Epistemological Issues in Embodied AI, 28:6, 499-520

[15] Franklin, S. 1997b. Artificial Minds, Cambridge, Mass: MIT Press.

[16] Glenberg, A., Havas, D., Becker, R., and Rinck, M. 2005. Grounding Language in Bodily States: The Case for Emotion. In The Grounding of Cognition: The Role of Perception and Action in Memory, Language and Thinking. Cambridge, UK: Cambridge University Press.

[17] Gottman, J. M., Murray, J. D., Swanson, C. C., Tyson, R., and Swanson, K. R. 2005. The Mathematics of Marriage: Dynamic Nonlinear Models, Cambridge, Mass: MIT Press.

[18] Greene, J.D. 2010, Neuropsychology and Ethics. American Psychological Association Conference, Boston, Mass.

[19] Greene, J. D. 2004. Why are VMPFC patients more utilitarian? A dual-process theory of moral judgment explains. Department of Psychology, Harvard University, Cambridge, Mass.

[20] Harnad, S. 1990. The Symbol Grounding Problem. Physica D, 42, 335-346.

[21] Horgan, T. and Tienson, J. 1989. Representations Without Rules. Philosophical Topics, 17 (Spring), 147-174.

[22] James, W. 1892, Psychology—The Briefer Course, Notre Dame: Indiana: University of Notre Dame Press 1985.

[23] Kalkanis, C. (last accessed June, 19, 2011) Affective Computing—Improving the performance of context-aware applications with human emotions. http://cswww.essex.ac.uk/Research/digital/CK_pres.pdf

[24] Lambrinos, D. and Scheier, Ch.. 1995. Extended Braitenberg Architectures. Technical Report, No. 95.10, AI Lab, Computer Science Dpartment, University of Zurich, Zurich, Switzerland.

[25] LeDoux, J. 2003. Synaptic Self: How Our Brains Become Who We Are, New York, New York: Viking

[26] Lewis, M. D. 2005. Bridging emotion theory and neurobiology through dynamic systems modeling. Emotional and Brain Sciences, 28(2), pp. 169-245.

[27] Lorini, E. 2008. Agents with emotions: a logical perspective, ALP Newsletter, Vol. 12, No. 2-3, August

[28] Neyem, A., Aracena, C., Collazos, C. A., and Alarcon, R. 2007. Designing Emotional Awareness Devices: What one sees is what one feels, Ingeniare, Revista chilena de ingenieria, Vol. 15 No 3, 227-235

[29] Panksepp, J. 1998, Affective Neuroscience: The Foundations of Human and Animal Emotions, New York, New York: Oxford University Press.

[30] Picard, R. 1997. Affective Computing, Cambridge, USA: The MIT Press

[31] Polyani, M. 1966, The Tacit Dimension, Chicago, Ill: University of Chicago Press, 2009.

[32] Popper, K. 1959, The Logic of Scientific Discovery, London, UK: Routledge, 2002.

[33] Russell, J. A., Bachorowski, J. A., and Fernandez-Dols, J. M. 2003. Facial and vocal expressions of emotion. Annual Review of Psychology, 54, 329-349.

[34] Russell, S. and Norvig, P. 2009. Artificial Intelligence, A Modern Approach, New Jersey: Prentice Hall.

[35] Scherer, K. R. 2009. Emotions are emergent processes: They require a dynamic computational architecture. Philosophical Transactions of the Royal Society of Biological Sciences, 364 (1535), 3459-3474.

[36] Searle, J. 1980. Minds, Brains, and Programs. Emotional and Brain Sciences, Vol. 1, 417-424.

[37] Strogatz, S. (1994), Nonlinear dynamics and chaos, New York: Perseus Publishing Company.

[38] Zhou, J., Yu, C., Riekki, J., and Karkkainen, E. 2007. AmE Framework: a Model for Emotion–aware Ambient Intelligence, The Second International Conference on Affective Computing and Intelligent Interaction (ACII2007): Lisbon, Portugal

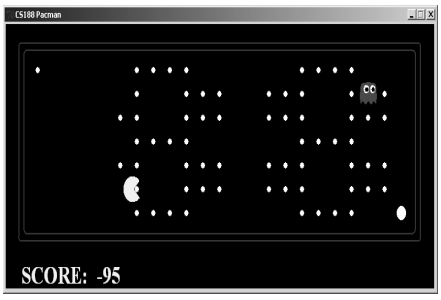


Figure 1. Snapshot from a sample run (Pacman is semi-circular, ghost is square, and food is oval)

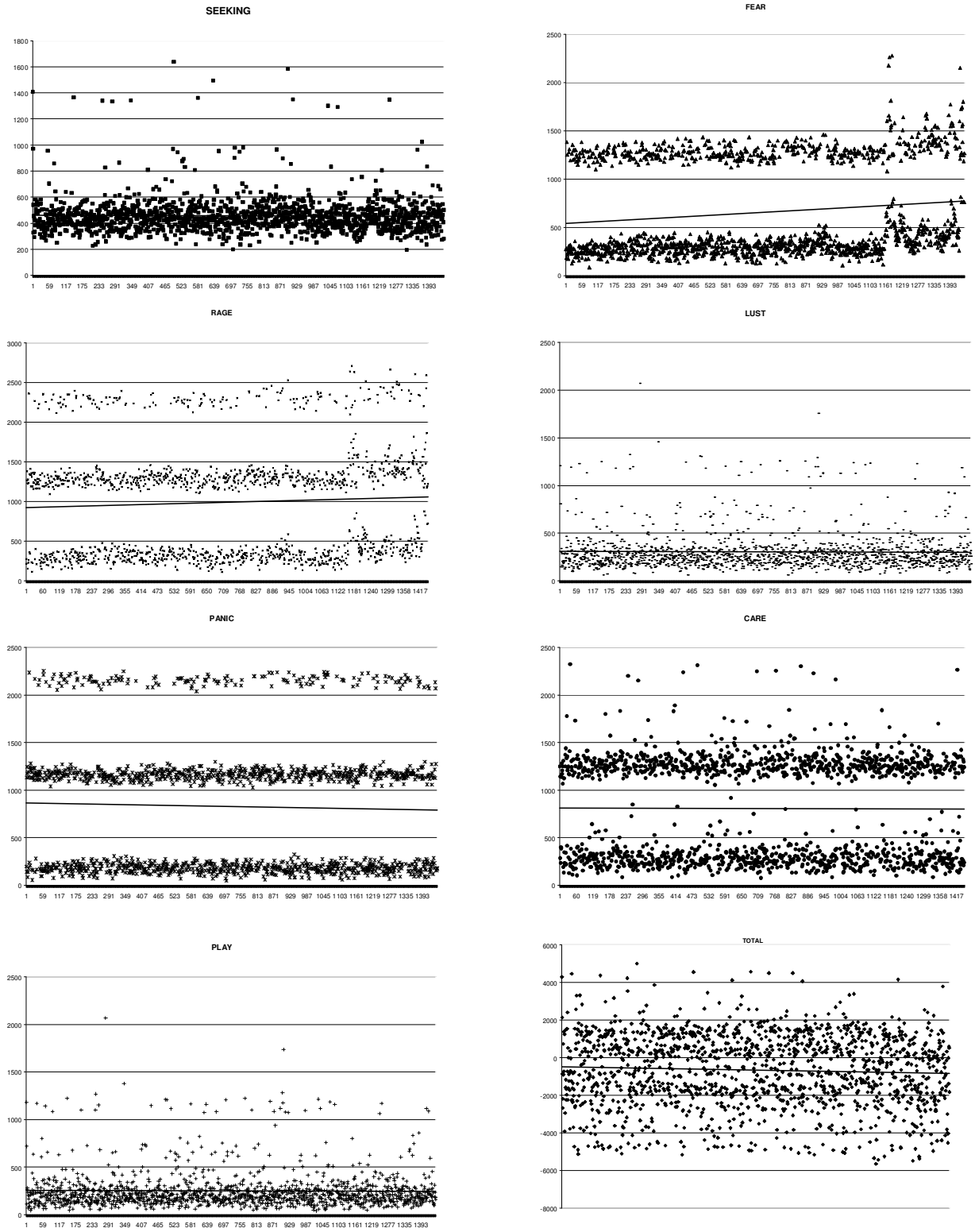


Figure 2. Affect values vs. iteration steps for the Panksepp variables modeled with a simple linear dynamical system

 ISBN: 978-1-61208-158-8

Towards an Adaptive Forecasting of Earthquake Time Series from Decomposable and Salient Characteristics

Simon Fong, Zhou Nannan
Department of Computer and Information Science
University of Macau
Macau SAR
ccfong@umac.mo, ma96578@umac.mo

Abstract—**Earthquake forecasting is known to be a challenging research program for which no single prediction method can claim to be the best. At large, earthquake data when viewed as a time series over a long time, exhibits a complex pattern that is composed of a mix of statistical features. A single prediction algorithm often does not yield an optimal forecast by analyzing over a long series that is composed of a large variety of features. A new analytic framework is proposed that allows these mixed features from the time series to be automatically extracted by a computer program, and fed into a decision tree classifier for choosing an appropriate method for the current forecasting task. The motivation behind this concept is to let the data decide which prediction algorithm should be adopted, adaptively across different periods of the time series. The contribution of this paper is twofold: (1) a framework of automatic forecasting which is very suitable for real-time earthquake monitor is proposed, and (2) an investigation on how different features of the data series are coupled with different prediction algorithms for the best possible accuracy.**

Keywords-Earthquake prediction; time series forecasting; automatic and adaptive forecasting; ARIMA; Holt-Winter's.

I. INTRODUCTION

In literature a large collection of time series forecasting algorithms have been studied and they have been applied for different domains such as finance [1], safety of power system [2], cargo volume [3], and traffic [4], etc. These algorithms show pros and cons in various situations, and mostly they were used for manual analysis where human experts were involved. Forecasting by manual analysis implied a full set of historical data or some intuitively selected length of the data series was used to be experimented by several ad-hoc forecasting algorithms, often trial-and-error or brute-force for the best candidate. This process is not only manual but usually would have to be carried out in the backend that falls short of real-time prediction capability.

Automatic forecasting is more desirable than backend analysis especially for critical scenarios, for instance, real-time monitors for earthquake prediction [5, 6], where fresh inputs from sensors are streamed in continuously and the responsive prediction must be made instantly if a fore-warning were to be made. This kind of real-time prediction scenario would often be handled by a computer program that is capable for processing a large amount of data, extracting characteristics from them and picking a suitable forecasting algorithm plus the required parameters, automatically.

Furthermore, earthquake data are multi-dimensional in nature, so are the prediction tasks. Earthquake data are spatial-temporal events; samples are measures of magnitudes, frequencies, depth, and they are collected from different places at different times. Aggregating up the measurements with other seismological and climate factors and time scales will multiply the dimensions, therefore demands for automatic forecasting for the sake of both speedy and accurate prediction results.

One major obstacle for achieving automatic forecasting is the choice of the appropriate time series forecasting algorithm and the associated parameters for the best results. In the mathematics domain, we are not short of theoretical models for forecasting; rather practically, how each of these different models could be dynamically chosen in run-time based on the characteristics of the latest section of the data series that is being analyzed – this forms the motivation of the research as in this paper.

On the other hand, earthquake data when formatted into a time series, it is found to possess a set of complex features which is often mixed into some composite along a very long range of time (>30 years), with no simple distinct characteristics. Consequently this implies various prediction techniques that have their strengths for different characteristics of patterns should be applied in turn adaptively for each segment of the time series, instead of fixing on a single algorithm that may deem fit at the start for the whole of the historical time series. The rationale of this adaptive prediction model is based on the observations that mixed features reside in earthquake patterns, these features are ever changing over time, and local trends are more relevant than global trends in earthquake. The second observation is still quite debatable because nobody can be certain that global trends have absolutely no effect on us given that the archival dataset only has a history of merely three decades. Local trends however are known to have temporal effect over a period of time and relevant to future events. For example, seismic activities are related in time and over some places, such as fore-shocks and after-shocks.

In this paper, we propose an adaptive forecasting framework for predicting the frequency of earthquakes in future time to come. The advantage of our framework is to facilitate automatic real-time forecasting. Section 2 reviews related works on automatic forecasting. Our proposed framework is described in details in Section 3. Section 4 shows our preliminary experiments. Section 5 concludes.

 ISBN: 978-1-61208-158-8

II. RELATED WORKS

In the literature, tons of forecasting algorithms and methodologies have been studied on univariate time series data from statistical perspectives. The methods are quite mature in terms of accuracy and robustness. Their applications are applied in a wide range of domains for solving real life problems. Automatic forecasting is a relatively unexplored area in software engineering that is comprised of different techniques, in addition to of course existing forecasting algorithms. The challenges of automatic forecasting basically can be divided into three conceptual levels: 1. Model selection, 2. Parameters selection for the chosen model, and 3. Data selection – how long the input data should be used in order to yield a result that has the highest accuracy? Given certain input data, how far ahead in future time intervals the forecast can sustain its accuracy?

With these three levels of complications in automatic forecasting, this section reviews some related works pertaining to practical challenges at each level.

A. Model Selection

Most researchers advocated all the popular forecasting models should be first trialed and the one that produced the lowest error is selected in an automatic forecasting approach. The merit of trying out all the models is apparently a way to guarantee the best prediction output, which is as good as using brute-force approach. The drawback of this method is the long computation time required that may be applicable to manual analysis where highest accuracy needs to be assured and an analyst can take a long time to find the perfect model. For real-time forecasting where a very short time constraint is imposed, some heuristic search for the optimal model is needed.

In the 70's, an epitome of using all the forecasting techniques in the hope of finding a technique that is best suited to a certain scenario is called M-Competition [7, 8] by Makridakis et al. It was concluded from M-Competition that statistically sophisticated methods might not necessarily outperform simple forecasting ones. In year 2000 the same authors focused on using only a limited set of exponential smoothing models, and they demonstrated that automatic forecasting can made to be particularly good at short term forecasts [9]; this methodology is extended to seasonal short-term series where the result beats all other methods.

Recently, Hyndman and Khandakar [10] improved this exhaustive M-Competition approach for finding the suitable method in automatic forecasting. They used state space models that underlay exponential smoothing methods for shortening the search process. And the authors also proposed a step-wise algorithm for forecasting with Autoregressive Integrated Moving Average (ARIMA) models.

Lemke and Gabrys in their work [11] suggested identifying an extensive pool of computable features from the time series for choosing a forecasting method. The judgmental feature selection used in [11] was a simple decision tree that decides which forecasting method should be used. Based on this concept, the author in this paper extended the decision tree into one that is for stream mining, treating the input time series data as a continuous stream; thus instead of preprocessing the full set of time series data for constructing (training) up the decision tree, our model progressively builds the decision tree ground up and updates the decision tree dynamically as each new segment of time series arrives. The advantage of using decision tree of stream mining is that the forecasting model can better adapt to different parts of the time series as supposedly the time series should never end but amounts to infinity, and the features of the series ever keep changing in values at different times. The adaptive model is hence suitable for real-time forecasting applications where forecast is made on the ever updating data stream.

B. Parameter Selection

Many time-series analysis techniques like regression analysis and exponential smoothing techniques require a set of parameters or constants to be initially chosen by users. The forecast accuracy largely attributes to the choice of these parameters at all times. Basically, there are two schools of researchers on two practices that deal with parameter selection. One common practice [12] is to utilize Solver (an optimization plug-in for MS-Excel) to simultaneously optimize the parameters and smoothing constants in spreadsheet prior to forecasting time series data. Rasmussen in his work [13] optimized using Solver initial or starting parameters as well. This work showed improved fits when the initial parameters and the smoothing constants are optimized together. The other approach [14] is to treat the update equation and the selection of parameters in the forecasting equations as a recursive updating function. The smoothing parameters are optimally chosen by a simple grid search according to some criterion. The solution may fall on local optima as the whole set of data are not globally considered.

In our proposed model, we combine the aforementioned approaches; because of the rolling nature of our forecasting process, the last optimized parameters from the previous segment of the time series would serve as the optimal initial parameters for the current segment by the concept of [13]. The parameters of the forecasting equations for each section of the time series are updated iteratively according to [14].

C. Data Selection

The selection of time scales as well as the length of time series has been studied widely in the literature. Generally forecast combinations that are based on segments of the series are said to be superior to their constituent forecasts. The rationale behind is that the conditions that vary along the time series, consists of trend and other statistical elements changes, hence the forecasting parameters drift. Previous works [15, 16, 17] show that it is feasible to do forecasting over different terms of the time series. Sets of forecasts from both short and long terms are merged to outperform either one. Cointegration is the core method adopted in piecing up the forecasts. This underlying concept of forecasting on individual segments instead of a full length of the time series drives the need of an adaptive forecasting model which continuously chooses the best suitable forecasting techniques and parameters on the fly over a train of time series segments.

 ISBN: 978-1-61208-158-8

III. FRAMEWORK FOR ADAPTIVE FORECASTING

A. *Methodology*

Adaptive forecasting is about a continuous forecasting process, which takes the time series input as an ever changing data stream, and automatically chooses a suitable forecasting method based on the characteristics of the recent data. The selection of the forecasting method is adaptive to the time series whose characteristics may vary across different observation time periods.

A classical forecast development process by Armstrong [16] was designed for analyst who makes once-off forecasts from a full length of time series. Our adaptive forecast however is derived from segments of time series of variable length as predefined by the user. The adaptive forecasting process can be implemented as a computer software program; hence human intervention can be relieved making it suitable for automatic and real-time prediction when the source data feed is properly setup.

The methodology of adaptive forecast process which is extended from [18] is shown in Figure 1.

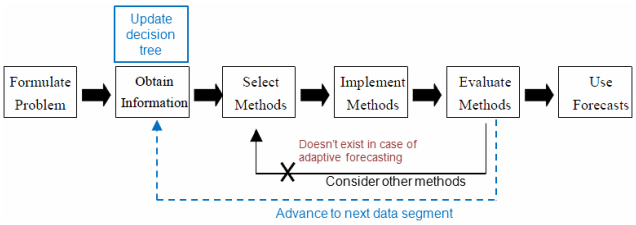


Figure 1. Adaptive forecasting methodology.

The main differences between Armstrong's process and the adaptive process are the use of a decision tree classifier in picking the suitable forecasting method and the time series is inputted to the classifier progressively in a stream of segments. In the original Armstrong's process, the time series is initially studied by identifying the potential explanatory variables. Yaffee [19] rendered the series stationary in mean, variance, and autocovariance; then next step was to manually select a set of forecasting methods from a pool of all available methods, probably by domain knowledge; tried them out, evaluated their performance. The loop repeated by trying the remaining methods if the earlier methods were not performing satisfactorily. In our methodology this loop is eliminated because the manual selection is replaced by a decision tree that instantly finds the appropriate method with optimized parameters given the statistics of the series segment.

The operation of the adaptive forecasting is iterating as the time series rolls along in segments into the classifier. Therefore fresh forecasts are obtained incrementally as time elapses based on the latest data segment. According to the study in [20], it is known that global trend models may not offer the best results. Some researchers advocate that forecasting models that tap on local trends can generally provide far better descriptions of real data. For instance, exponential smoothing algorithms give preference to recent trends by exponentially fading off the weights of past factors.

B. *Streaming Operation*

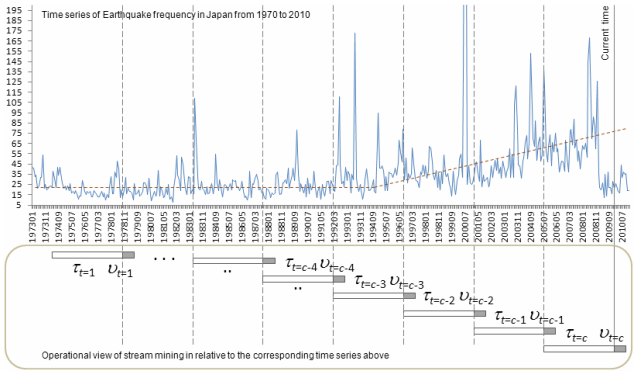
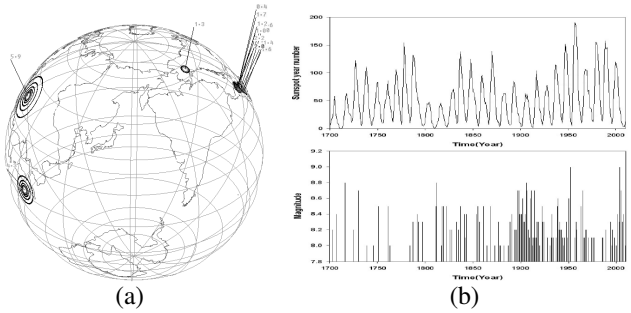


Figure 2. Operational view of adaptive forecasting process – the time series is being forecasted in the fashion of sliding-window.

As shown in Figure 2 above, the adaptive forecasting process operates in the form of data streaming that could be briefly described in the following sequence of steps:

1. The time series is arriving in segments each of which has a certain length, $|\mathcal{A}|$ = size of observation period. The length is a user-defined variable that should be calibrated in advance. When the length is too long, the forecast will be as good as using a global trend and time lags would exist in between each pair of successive forecasts. If the length is too short, the forecasting accuracy degrades due to insufficient amount of past data were used. In regression, some suggested the minimum length to be the number of cycles equals to one plus the number of coefficients.
2. With each segment of time series data under observation, extract a comprehensive set of salient characteristics (to be explained in the next section). While most of the characteristics would be used for training up the forecasting techniques selector, some useful salient features can be used for other data analysis such as outlier detection, intra-correlation of earthquake events, and inter-correlation between earthquakes and other external activities. Some examples are shown in Figure 3.



(a) (b)

Figure 3. Examples of other analysis by using the statistical characterstics of Earthquake time series: (a) Intra-correlation between earthquake events that happen at the same time across different locations of the Earth, (b) comparison of earthquakes of magnitute > 8 and the number of sun spots.

 ISBN: 978-1-61208-158-8

3. A real-time decision tree classifier which is implemented by Hoeffding Tree [21] algorithm updates itself with the given salient characteristics and the feedback errors from the previous evaluation. The decision tree chooses a suitable forecasting method automatically upon the arrival of a new data segment. The fundamental reason for using Hoeffding Tree is its real-time ability in handling high-speed data stream; its decision tree model can be updated (refreshed) almost instantly whenever a piece of fresh data arrives. In contrast, traditional decision tree requires reading up the whole set of past records plus the update for retraining the model.
4. Once a forecasting method is chosen, it predicts the future points by computing over the observation points within $|\tau|$. $|\upsilon|$ is the length of forecast-ahead period; once again, it is a predefined variable by user. In the example shown in Figure 2 which consists of a time series of earthquake frequencies occurred in Japan from 1973 to 2010, υ/τ is 0.1 where υ=6 months for testing and τ=66 months for training. So the ratio is approximately 0.1 that is common in data mining for training and testing a decision tree. Generally it is known that the accuracy will drop as the forecast-ahead period increases and large observation points are needed for a far-ahead prediction.

Without exhaustive mathematical proofs, streaming operation for adaptive forecasting has its advantage over traditional forecasting from a global trend by looking at the example in Figure 2 intuitively. Represented by a dotted line along with the time series, the linear regression trend is divided into two distinctive eras. Prior to year 1993 the time series, as shown by the real data of earthquake frequencies that took place in Japan obtained from USGS, exhibits a relatively flat gradient. The frequency of earthquake, however, started to climb almost steadily from 1993 onwards. Consequently this observation is indicating a fact that the trend of increase in earthquake frequency suddenly picked up in 1993. And this fact is telling us that the use of trend information (as a salient feature) must be adaptive in our forecasting method. That is, the trend information should be observed periodically and only the most updated information akin to local trend should be used for forecasting. As a counter-example if the global trend that embraces the series from 1973 to 2010 is used for forecasting future events from now on, the predicted values would likely be underestimated because there was indeed an up shift in the trend in 1993.

The adaptive forecasting process ensures the characteristics of the time series are always up-to-date, and the appropriate forecasting method is used based on the current values of the salient characteristics. This fundamentally changes the forecasting paradigm or tradition, from examining the whole series by manually chosen forecasting methods, to a new dynamic and adaptive streaming manner that facilities real-time and accurate forecast.

C. *Decomposing the Time Series into Salient Features*

Salient characteristics in the context of time series forecasting are generally known to be statistical features that spur out beyond their nearby data points. Visually they are the patterns that are noticeable, outstanding, and prominent or can be just easily identified in relative to other parts of the time series. Descriptive statistics are usually the mechanisms for generating these salient characteristics from the time series data. In the adaptive forecasting methodology, decomposing the salient features from the time series is a task of Obtain Information. The salient features as components of a time series can be extracted include but not limited to those listed below:

TABLE I. SOME USEFUL SALIENT FEATURES FOR OUR MODEL

Salient features	Description
Level	Average value of the time series, (mean, variance)
Trend	A reasonably long-term sweep or general direction of movement in a time series
Seasonality	Short-term cyclical behavior that can be observed several times
Noise	Random variation
Outlier	Outstanding values, known as white noise
Auto-correlation	Cross-correlation of a time series with itself, measured in autocovariance
Stationarity	Measure of a stochastic process by its change of joint probability distribution. Sometimes measured by stationary transition probabilities
Random walk	A phenomenon that a time series changes by the same probability distribution and the movements are independent of each other. In this case the past movement of a data point fails to predict its future movement. The data points take a random and unpredictable path.

In addition to those listed above which were used in our experiment, there are many other potential salient features derived from the time series that may relate to selection of the best forecasting method: residuals from previous prediction, causal forces, consistent trends, contrary series, damped trends, decay forces, decomposition, discontinuities, extrapolation, growth forces, inconsistent trends, instabilities, opposing forces, regressing forces, reinforcing series, start-up series, and supporting forces etc. There are also other tests and procedures like Anderson Darling normality test, information derived from Box pierce test, histogram, and QQ plot which shall be exploited in this project. Other factors that might influence the accuracy of the final forecasting result pertain to the streaming operation of our model, such as the choices of τ, the observation period and υ, the prediction period ahead.

According to [22] it is essential to know the salient characteristics of the time series prior to choosing a suitable forecasting method, especially for exponential smoothing method that largely based on the trend and seasonality of the past data points to predict the future ones. In the work [22] it was already shown that the exponential smoothing forecasts

 ISBN: 978-1-61208-158-8

can become more accurate, if the better the diagnosis of the salient characteristics was done on the time series.

D. Decision Tree As Forecaster Selector

Extended from Box-Jenkins methodology which finds suitable types of autoregressive (AR) and moving average (MA) techniques from the salient features of the past values of the time series to make forecasts, our model flexibly covers a wider range of techniques. The underlying assumption for the adaptive model is that the time series is composed of a mix of salient features which shall not be forecasted very accurately by any single model alone, and these salient features change their values across time. Therefore a flexible scheme is needed which does not only select the best forecaster automatically, the selection scheme can have learning ability that dynamically fine-tune the 'decision rules' by itself as time goes.

In order to support such adaptive forecaster selector, a light-weight decision tree based on Chernoff-Hoeffding bounds from stream-mining [23] is adopted as a core engine for implementation. Readers who want to obtain details about the decision tree are referred to [23]. The design of the decision tree is quite standard, default parameters are used. The main advantage of the so-called light weight decision tree is its adaptive ability to readjust its structure (rules as tree branches) when input data stream is feeding in. In other words, the classifier is able to adapt to the changing salient characteristics of the time series as new samples of time series are coming in, and it will always choose the relative suitable forecasting method for the new samples.

The rules in the decision trees need to change though at the start the basic tree structure is predefined (pre-trained) according to the experiences learnt from our experiments. The branching conditions at the tree nodes are more or less floppy in a sense that the values are numeric and should be adjusted by the learning experience in the form of error (residual) feedbacks. For example, IF Trend=*strong*, THEN Regression_Model is chosen; How strong then is *strong*? We started the decision tree with default values, and let the data and forecast errors do the fine-tuning – the rules will be periodically updated by reassessing the performance.

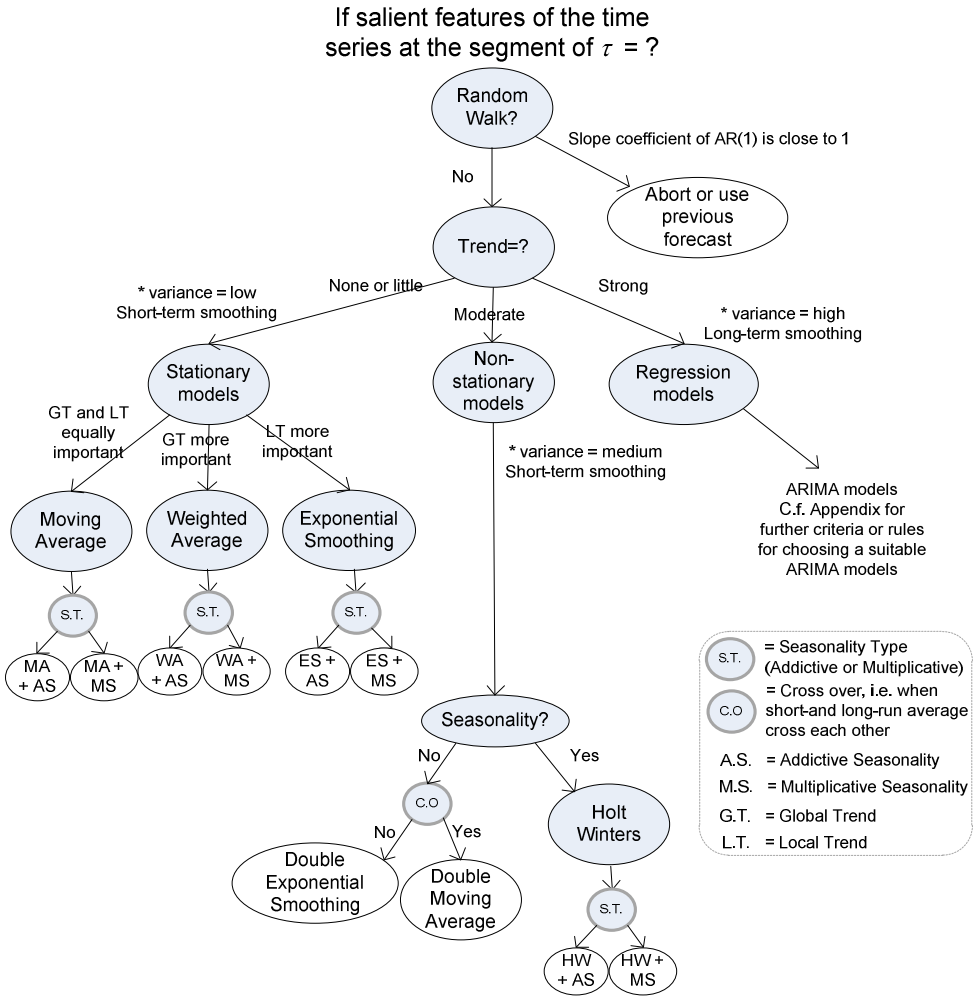


Figure 4. Decision tree for deciding which forecasting method to use, based on the salient characteristics of the time series that are currently under observation.

As can be seen from the decision tree in Figure 4, the very first thing to determine whether the segment of time series is predictable in a sense that its past can predict its future or not. Such predictability of the series can be evaluated by checking the behavior of random walk. A random is a series in which changes from one time period to the next are completely random. It is a special case of AR(1) model in which the slope coefficient is equal to 1. In our experiment of Japan earthquake frequency time series, the AR(1) model has a slope coefficient of 0.5722 and standard error 0.0383 that is more than 10 standard errors away from 1, signifying that the series is not a random walk. (In contrast a financial series of S&P500 monthly closing prices between May 1995 and August 2003 has a slope coefficient of 0.985 and a standard error 0.015 [24]).

Since the predictability of our earthquake data is validated, the decision tree then proceeds to select a suitable forecasting method. From experiences, as shown in our decision tree, trends, stationarity and seasonality of a series are significant factors for choosing a prediction method. In general the higher the node or the conditions for the branching split is the more decisive power it has in the decision making. Nevertheless the example in Figure 4 shows a decision tree that is constructed with conditions of the main salient features. There are other salient factors, perhaps slightly less important are not shown but they are there to further extend out the tree with more conditions down at the branches.

IV. EXPERIMENTS

Forecasting experiments are run by using our proposed adaptive model and other well-known forecasting methods alone, in order to validate the efficacy of the new model.

The time series data we used were real data originally downloaded from USGS archive. The raw data are historical records of earthquakes that happened in Japan from January 1973 to December 2010 of magnitudes of all levels. For the sake of general interest, the data are transformed into frequency of earthquake per month, with Richter magnitude of at least scale 3. The data has a mean frequency 36.09 per month, standard deviation 33.22, maximum 482 and minimum 8 times per month respectively. By just visually inspecting the time series as shown in Figure 2, the data has a complex mix of salient characteristics; it is somewhat stationary with a slow shift of a rising trend, obviously at the pivot year of 1993, and a sharp drop at the end of the year 2008. A new trend seems to develop from 2008 afterwards, and apparently this local trend is quite unknown due to the shortage of data between 2008 and current year (2011). The seasonality of the time series, as shown in the ACF Plot of autocorrelations in Figure 5 has a set of values exceeding the UCI and LCI, indicating that there is a substantial extent of seasonality though not very strong. The seasonal cycles lack of a strong regularity too as seen in the plot. At the decision tree this salient characteristics should fall upon Holt-Winter model and occasionally into autoregression models as the applied methods. The PACF Plot of partial autocorrelations suggests that the seasonal data depend on other external factors most of the time as well.

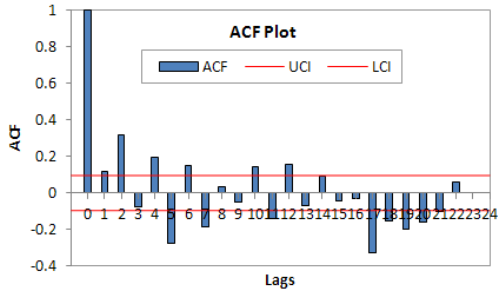


Figure 5. ACF Plot of the earthquake frequency time series

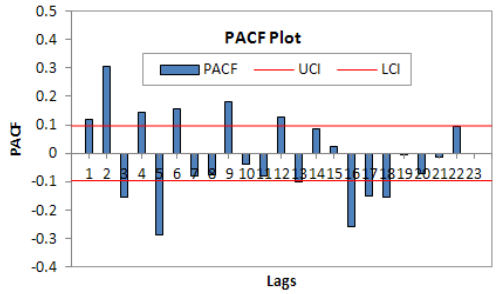


Figure 6. PACF Plot of the earthquake frequency time series

Preliminary results were obtained from running the proposed adaptive model based on the decision tree specified in Figure 4. In general the adaptive model shows its advantage in finding a suitable model therefore producing supposedly the best available result. As observed from the preliminary results from Table 2, the adaptive model can achieve a slightly higher accuracy than most of the other methods which are applied alone, except when there are large fluctuations and sudden change in the patterns. This suggests that additional salient features other than those shown in Figure 4 in the decision tree ought to be used.

TABLE II. COMPARISON OF ADAPTIVE MODEL AND OTHERS ALONE

Forecast type	Average accuracy over all the segments
Holt-Winter	81.40%
Addictive HW	87.20%
Multiplicative HW	80.20%
ARIMA	70.40%
Moving Average	76.70%
Adaptive Model	85.10%

Our adaptive model was probably impaired by the mis-classification by the trend strength into ARIMA. ARIMA was under-performed in time series forecasting in this case because the trend was not clearly strong and the trend might have badly affected by the outlier. A snapshot of forecasting performance as shown in Figure 7 sheds some light on the causes. Evidently the forecast by ARIMA under predicted most of the peaks especially the large ones – outliers. The magnitude of under prediction was almost by half in year 2000 July where Japan had an exceptional large number of earthquakes (482 times). Also the model showed a consistently long period of over-prediction throughout year 2010, because of the sudden and sharp change of trend.

 ISBN: 978-1-61208-158-8

V. CONCLUSION AND FUTURE WORKS

In previous studies combined forecasts have generally been shown to outperform the forecast from the single best model. It is generally known to many researchers that no single forecasting method can always yield the best results. Therefore selection of the appropriate forecasting methods for different time series has been a popular research topic. Most of the works in the past however concentrated on defining certain fixed rules or guidelines in choosing the best forecasting method. The full time series was often considered under this forecaster selector by diagnosing its salient characteristics. This automatic selection of forecasting methods was extended and pondered on in this paper, with the aim of designing an adaptive model that can be applied in real-time forecasting while the time series was analyzed on the fly. A dynamic decision tree from stream mining was applied in the new model which is able to adjust the rules as reflected by the structure of decision tree. The time series is analyzed in consecutive segments, each of which the salient characters are extracted and used for the decision tree to decide on the forecasting method. An experiment was conducted by taking the live earthquake data from Japan. Importantly this particular time series shows a good example of a complex series that embrace of different salient characteristics at different times; therefore, intuitively no single method should be used to forecast the whole time series since different forecasting method has its strength and weakness for different values of salient characteristics.

The contribution of this paper is a framework and methodology that can be programmed in an automated system for continuous forecasting, such as earthquake pre-warning system, for instance.

REFERENCES

[1] Yaohui Bai, Jiancheng Sun, Jianguo Luo, and Xiaobin Zhang, "Forecasting financial time series with ensemble learning", 2010 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), 2010, pp. 1-4.

[2] Ying Jiang, Yicheng Ye, and Qin Wang, "Study on Weighting Function of Weighted Time Series Forecasting Model in the Safety System", 2011 Asia-Pacific Power and Energy Engineering Conference (APPEEC), 2011, pp. 1-4.

[3] Xiao Shengling, Wang Wei, and Wang Bo, "Study on forecasting method of highway port cargo volume", 2010 International Conference on Logistics Systems and Intelligent Management, Volume 3, 2010, pp. 1831-1834.

[4] Shen Fu-Ke, Zhang Wei, and Chang Pan, "An Engineering Approach to Prediction of Network Traffic Based on Time-Series Model", JCAI 2009 International Joint Conference on Artificial Intelligence, 2009, pp. 432-435.

[5] Kawaguchi, K., Kaneda, Y., and Araki, E., "The DONET: A real-time seafloor research infrastructure for the precise earthquake and tsunami monitoring", OCEANS 2008 - MTS/IEEE Kobe Techno-Ocean, 2008, pp. 1-4.

[6] Ji Chang-Peng and Liu li-li, "Real-time Detection for Anomaly Data in Microseismic Monitoring System", International Conference on Computational Intelligence and Natural Computing, (CINC 2009), Volume 2, 2009, pp. 307-310.

[7] Makridakis S., Hibon M., "Accuracy of forecasting : an empirical investigation with discussion", Journal of the Royal Statistical Society A, Volume 142, 1979, pp. 97-145.

[8] Makridakis S., Chatfield C., Hibon M., Lawrence M., Mills T., Ord K., and Simmons L. F., "The M2 competition : a real-time judgmentally based forecasting study", International Journal of Forecasting, Volume 9, 1993, pp. 5-23.

[9] Makridakis S. and Hibon M., "The M3-Competition: Results, Conclusions and Implications", International Journal of Forecasting, volume 16, 2000, pp. 451-476.

[10] Hyndman R. and Khandakar Y., "Automatic Time Series Forecasting: The forecast Package for R", Journal of Statistical Software, Volume 27, Issue 3, July 2008, pp. 1-22.

[11] Lemke C. and Gabrys B., "On the benefit of using time series features for choosing a forecasting method", Proceedings of the 2nd European Symposium on Time Series Prediction, Porvoo, Finland, 2008.

[12] Ragsdale C. and Plane DR., "On modeling time series data using spreadsheets", Omega 2000, The International Journal of Management Science, Elsevier, 2000, Volume 28, Issue 2, pp. 215–21.

[13] Rasmussen R., "On time series data and optimal parameters", Omega 2004, The International Journal of Management Science, Elsevier, 2004, Volume 32, pp. 111–120.

[14] Gelper S., Fried R. and Croux C., "Robust forecasting with exponential and Holt-Winters smoothing", Technical report KBI 0718, Faculty of Economics and Applied Economics, Katholieke Universiteit, Leuven, April 2007, pp. 1-22.

[15] Engle, R. F., Granger, C.W. J., and Hallman, J. J., "Merging short-and long-run forecasts: An application of seasonal co-integration to monthly electricity sales forecasting", Journal of Econometrics, Volume 40, 1989, pp. 45-62.

[16] Casals, J., Jerez, and M., Sotoca, S., "Modelling and forecasting time series sampled at different frequencies", Journal of Forecasting, Volume 28, 2009, pp. 316-342.

[17] Granger, C. W. J., "Implications of seeing economic variables through an aggregation window", Ricerche Economiche, Volume 47, 1993, pp. 269-279.

[18] Armstrong, J. S. (Ed.) (2001). Principles of Forecasting: A Handbook for Researchers and Practitioners. Boston: Kluwer.

[19] Yaffee, R. A. (2000). Introduction to Time Series Analysis and Forecasting with Applications of SAS and SPSS. San Diego: Academic Press.

[20] Newbold P. and Bos T., "On exponential smoothing and the assumption of deterministic trend plus white noise data-generating models", International Journal of Forecasting, Volume 5, Issue 4, Elsevier Science, 1989, pp. 523-527.

[21] Fong S. and Yang H., "Enabling Real-time Business Intelligence by Stream Data Mining, Book Chapter of "Data Mining", Kimito Funatsu (Ed.), ISBN: 978-953-307-547-1, Intech, 05 January 2011, Vienna, Austria.

[22] Fomby T., "Exponential Smoothing Models", Technical report, Department of Economics, Southern Methodist University, June 2008, pp. 1-23.

[23] Yang H. and Fong S., "Investigating the Impact of Bursty Traffic on Hoeffding Tree Algorithm in Stream Mining over Internet", The 2nd International Conference on Evolving Internet (INTERNET 2010), September 2010, Valencia, Spain, pp. 147-152.

[24] Shmueli G., Patel N., and Bruce P., "Data Mining for Business Intelligence", Wiley, 2nd Edition, 2010, pp. 332.

[25] Arsham H., "Time-Critical Decision Making for Business Administration", Online: http://home.ubalt.edu/ntsbarsh/stat-data/Forecast.htm, 2003, [last access: 1/8/2011].

 ISBN: 978-1-61208-158-8

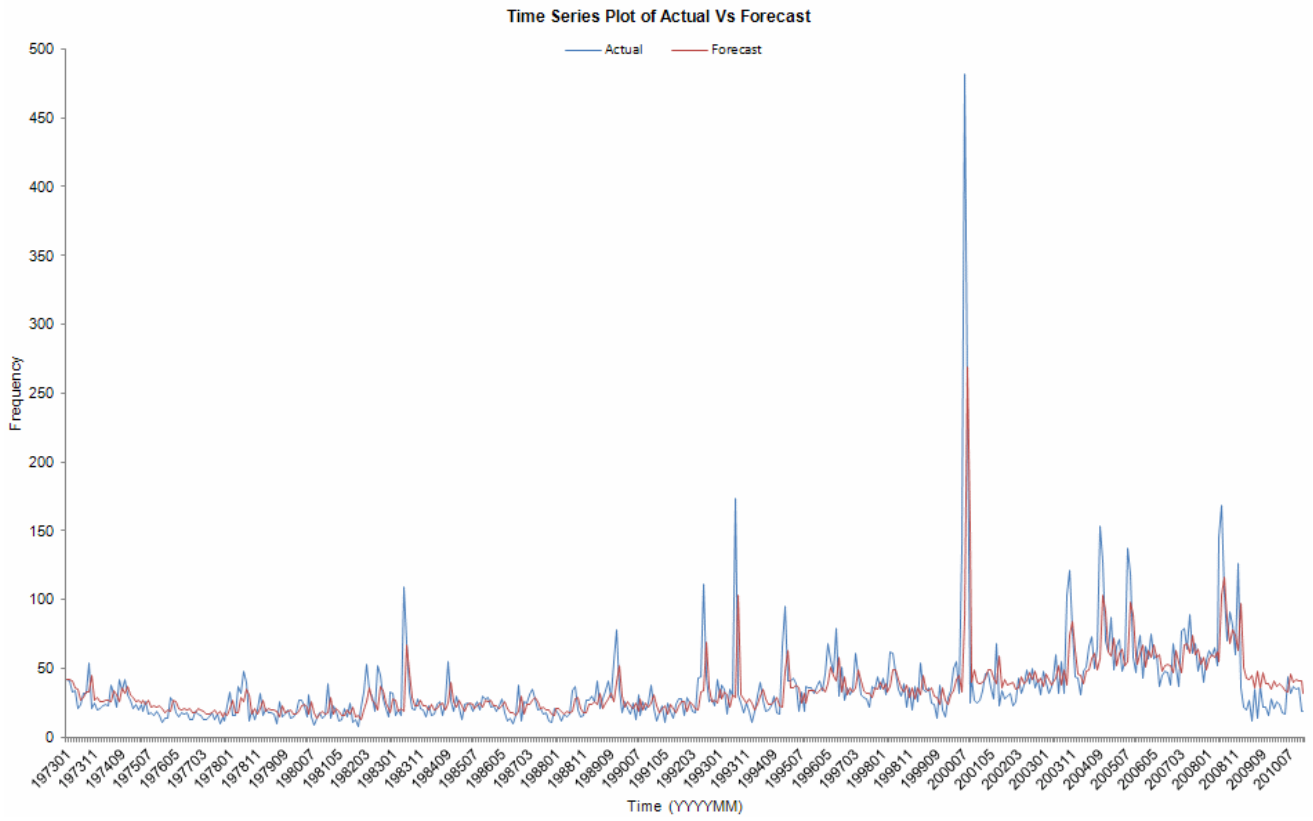


Figure 7. Time series plot of earthquake frequncies that took place in Japan from 1973 to 2010; the forecasting method used is ARMIA(1,1,1)

Appendix - Autoregressive Model Selection Criteria (Source: [25])

1. If none of the simple autocorrelations is significantly different from zero, the series is essentially a random number or white-noise series, which is not amenable to autoregressive modeling.

2. If the simple autocorrelations decrease linearly, passing through zero to become negative, or if the simple autocorrelations exhibit a wave-like cyclical pattern, passing through zero several times, the series is not stationary; it must be differenced one or more times before it may be modeled with an autoregressive process.

3. If the simple autocorrelations exhibit seasonality; i.e., there are autocorrelation peaks every dozen or so (in monthly data) lags, the series is not stationary; it must be differenced with a gap approximately equal to the seasonal interval before further modeling.

4. If the simple autocorrelations decrease exponentially but approach zero gradually, while the partial autocorrelations are significantly non-zero through some small number of lags beyond which they are not significantly different from zero, the series should be modeled with an autoregressive process.

5. If the partial autocorrelations decrease exponentially but approach zero gradually, while the simple autocorrelations are significantly non-zero through some small number of lags beyond which they are not significantly different from zero, the series should be modeled with a moving average process.

6. If the partial and simple autocorrelations both converge upon zero for successively longer lags, but neither actually reaches zero after any particular lag, the series may be modeled by a combination of autoregressive and moving average process.

 ISBN: 978-1-61208-158-8

Introducing new Pattern Language Concepts and an Extended Pattern Structure for Ubiquitous Computing Application Design Support

René Reiners
Fraunhofer Institute for Applied Information Technology FIT
Sankt Augustin, Germany
rene.reiners@fit.fraunhofer.de

Irina Astrova
Institute of Cybernetics
Talinn University of Technology
Talinn, Estonia
irina@cs.ioc.ee

Alfred Zimmermann
BIRC - Business Informatics RC
Reutlingen University
Reutlingen, Germany
alfred.zimmermann@reutlingen-university.de

***Abstract*—The method of collecting and communicating design knowledge in the shape of design patterns is a proven method, which is applied in different domains originating from architecture over software engineering, organizational aspects up to application design. This work introduces an extension of the pattern language concept and the applied pattern format introduced by Christopher Alexander, Jan Borchers and others. Our intention is to formulate a generic pattern language for ubiquitous computing application design. The language should enable contributors to integrate ideas and approaches from related domains into one pattern language construct. We present new and extended features for the "traditional" pattern language concept and its pattern structures on a conceptual basis. We also address needed features when opening the pattern language concept for a community that is asked to provide feedback to given patterns and contribute new findings.**

***Keywords*-Design Patterns, Pattern Language, Extension, Community, Collaboration**

I. INTRODUCTION

Mark Weiser's vision of *ubiquitous computing (UbiComp)* has motivated and influenced the development and implementation of smart embedded devices [1]. In the recent decades, their physical size has been significantly reduced accompanied by novel interaction concepts for smart environments. Currently, different industrial and research approaches enter different distribution channels creating a huge diversity of concepts concerning application design, service provisioning, and interaction techniques. The knowledge and experience from these approaches, however, is currently inherent to the individual systems and concepts or focusing on specific problems, respectively. Thus, general design guidelines derived from proven realizations are hard to reveal, capture, and transfer to other application designs.

As a consequence, proven concepts are currently hard to compare or combine. If the latter was possible, recommendations and ratings for often used combinations of good design practices could be realized. Well working concepts and designs are usually elaborately documented. Bad practices or failing concepts can also be valuable to avoid design mistakes a priori. There is a manifold of possibilities about what to offer and how to offer smart services in UbiComp environments [2]. Accessing these services and the way of interacting with them can also be very different. Examples for smart services offered in a UbiComp environment can be found in [3], [4], [5], [6], [7], [8]. Currently, smart services of different kinds have found their way to mobile devices. Mostly, they are independently encapsulated into small applications for current smart phones.

Looking at the potential for new ways of service provisioning and interaction, there is a danger for the users in this new world: *Disorientation*. In a new environment, how should they know what kind of smart services is available and how to access them? Maybe there is already experience with another system. However, this knowledge does not help in the current situation. Therefore, solutions need to be found that are generally applicable to many different situations and kinds of services and applications, respectively. This work seeks for methods to structure application design knowledge from current as well as future approaches and applications in the domain of ubiquitous computing. The structure is intended to serve as a repository containing guidelines supporting application designers in the domain of ubiquitous computing environments. Additionally, design flaws are rarely published, leaving the danger that they can potentially be repeated. This kind of design knowledge also needs to be preserved and made available together with working design guidelines. So, the central question of our work is formulated as follows:

"How can we abstract, combine, and keep alive UbiComp application design knowledge?"

II. PATTERN LANGUAGES AS APPROACH TO A SOLUTION

In [2], we described first steps towards an approach to gather, generalize, structure and manage UbiComp application design patterns. The first step was to find a common denomination of concepts that are used in different approaches and domains but still have common roots. Smart Services, Smart Devices and Smart Environments were introduced as well as an optional *take-away attribute* for smart services

 ISBN: 978-1-61208-158-8

allowing for remotely controlling smart services that are normally bound to objects and locations in the real world.

After the generalization of terms and concepts in application design, we need to find a *structure* for design practices. *Pattern Languages* provide a very flexible and extensible way to describe proven design solutions (cf. section III). The presented conceptual basis allows for constructing a first subset of patterns and the relations between them.

III. Related Work

Recurring design problems are a well-known phenomenon affecting many different domains. Not only in the technical sector but also in design areas like architecture, construction or user interface design, people see themselves faced with problems for which they do not directly know a solution.

However, it may happen that someone else has also treated the same or a similar problem and that a solution or guideline was found. In a good case scenario, this best practice was written down or kept in a way that it could be shared and distributed. Thus, a pattern represents knowledge for a specific domain, discusses the context in which it could be applied and explains ways to solve a certain problem.

Current pattern collections originating in the field of architecture [9], and being extending predominantly by Borchers in interactive exhibit design [10], up to the design of websites [11] or HCHI application design [12] mostly go even further. On the one hand, they can be distinguished by the vocabulary used for explaining a solution (e.g., very technical vs. descriptive and better suited to non-domain experts). On the other hand, many collections organize and structure particular units of information in a way that they can be organized hierarchically thus having different degrees of information detail.

Hierarchies, for example, can make use of the spatial dimension [9] or the pattern's level of technical description detail [12]. Others remain on a very technical level but then structure the patterns by purpose like in Gamma et al. in their collection of software design patterns [13].

Besides this informal way of describing design solutions, semiformal and formal approaches were developed. The three different concepts are briefly discussed in the following.

A. Informal Representation of Design Patterns

HCI and application design patterns are traditionally represented in natural language and stored in loosely coupled documents. Although there are different formats for such documents (the most common is a canonical form), the documents usually contain a set of fields describing the context in which a problem occurs, the intent of the design pattern, the relevant forces that justify why the design pattern and its implementation should be used and how to generate the solution for the problem. The term description seems more suitable for this type of representation.

Although informal representation helps users to understand the rationale behind a design pattern, there is a limitation to precision due to the use of natural language and the semantic ambiguity [14]. This disadvantage does not allow for any level of automation to resolve widely recognized problems such as finding and selecting the appropriate design patterns or applying them [15].

B. Semiformal Representation of Design Patterns (UML)

Several attempts have been made to represent design patterns in a semiformal way. Most of these attempts [15], [16], [17], [18] are based on UML (Unified Modeling Language [19]) descriptions. UML helps in specifying the structural and behavioral aspects of design patterns using class/object and sequence/collaboration diagrams, respectively. However, UML does little to help users understand the intent, applicability and consequences of design patterns [15].

C. Formal Representation of Design Patterns

As an attempt to resolve the problems above, different approaches were proposed to formalizing representation of design patterns. These approaches are based on either pure mathematics (i.e., first order logic, temporal logic, high order logic, object-calculus, sigma calculus, p-calculus, etc.) or ontologies [20]. The main goals of formalization are:

- To provide better understanding of design patterns and their composition; it helps to know when and how to use patterns properly in order to take advantage of them.
- To resolve issues regarding relationships between design patterns; it is not only important which design patterns are used to solve a given problem, but also it is important in which order they are applied.
- To allow for the development of tools that support activities regarding design patterns; these activities include semantic search for design patterns, automatic code generation and formal validation of design patterns.

1) Mathematics-based approaches: For example, Cornils and Hedin [21] described design patterns using reference attributed grammars with syntactic and context-sensitive rules. Eden et al. [22] derived LePlus (LanguagE for Patterns UniformSpecification) from Higher-Order logic to represent design patterns as logic formulas, which consist of the elements of object-oriented language (i.e. classes, methods or hierarchies) and relationships between them. Smith and Stotts [23] extended sigma calculus, which defines relationships between the elements of object-oriented language to describe design patterns.

Finally, Taibi and Ling Ngo [24] specified the structural and behavioral aspects of design patterns using first order and temporal logics, respectively. This approach described patterns in terms of classes, attributes, methods, objects and untyped values, and relationships between them.

 ISBN: 978-1-61208-158-8

The approaches adopted mathematics to describe design patterns. However, this can turn into a big disadvantage if users lack mathematical skills and thus, cannot easily understand how patterns have been described. Another big disadvantage with these approaches is that they describe the pattern solution only.

2) Ontology-based approaches: In recent years, several attempts have been made to formalize design patterns using ontologies, thus making the design patterns understandable for both humans and machines - this is the same idea as used in the scope of the Semantic Web[25]. For example, Rosengard and Ursu [15] introduced the idea of representing patterns as ontologies with a view to the development of tools for the automatic organization, retrieval and explanation of reusable solutions to software development, codes of good practice and company policies.

IV. Addressing a Larger Application Domain

Existing pattern languages are specialized on a closed problem domain allowing them to provide detailed knowledge for a certain class of design problems. However, we want to cover more than one aspect or problem domain and therefore search for ways to increase the extensibility of a pattern language. This also implies that not all formulated pattern can be chosen during the whole design process since readers have to decide for certain paths. Another point we want to address is the vividness of a pattern language. These new requirements were formulated in preliminary work on which we built in this work (cf. [2], [26]).

Analyzing related work, it can often be found that a group of authors defines a set of patterns from their experience and interlink them within their pattern language construction. However, in case that a pattern looses validity or is refuted by other readers who applied it, there is no process to feed back the experience into the pattern language. The approach resembles more a"try out and see" approach. Additionally, new ideas and found concept also cannot be included into the existing structure since there is no way for adding new nodes except for trying to include them together with the original authors in a new edition of the mostly written publication or by defining (yet) another custom pattern language. The new concepts and our proposed modification in the traditional pattern structure are described in the next chapter.

V. Introducing New Concepts

The pattern language approach described in this work extends the present approaches to functions that enable the construction of a collaborative community platform that we refer to as *Pattern Management System (PMS)*. With the help of this system, working application design approaches can be extended, modified and discussed.

In order to realize the PMS that also manages rules and processes for contributions, we decide for presenting the informal pattern language concepts (cf. III-A) to the reader. For the technical realization we need to consider at least semi-formal mechanisms to handle the pattern collection. In order to meet our requirements concerning a wide application domain and the mentioned community mechanisms, we extend the "traditional" pattern language concepts and pattern structures as described in the following sections.

A. New Concepts for a Pattern Language

The pattern language structure described in this work will follow the general concepts as described by [10] and [12] (cf. section III), but will also introduce new features resulting from the demand of covering a wider field of applications in the UbiComp domain as well as the support for external authors' contributions (cf. Figure 1 for illustrations).

Branching: While browsing the patterns, we introduce decision nodes at which the reader needs to decide for a subset of design recommendations. The decision may be based on interaction style, privacy demand or, e.g., device footprint. After making a decision, other subsets of design patterns are no longer available since the criteria for the intended design are not met any longer. This leads to a dynamic graph structure of causally related patterns.

Sequencing and Recommendation: Patterns that are located on the same level within the hierarchy represent design alternatives that can be chosen but also be implemented in parallel.

For the reader, it may be interesting which patterns where followed the most after deciding for certain predecessors and therefore providing a safe route of patterns to apply for a proven design pattern combination. Sequencing may be inferred from most read patterns or proposed combinations by others as well as on rankings. That way, different alternative paths can also be weighed by recommendations given by the community or by (semi-) automated proposals.

Community Involvement: The pattern language approaches discussed so far all relied on the knowledge of one

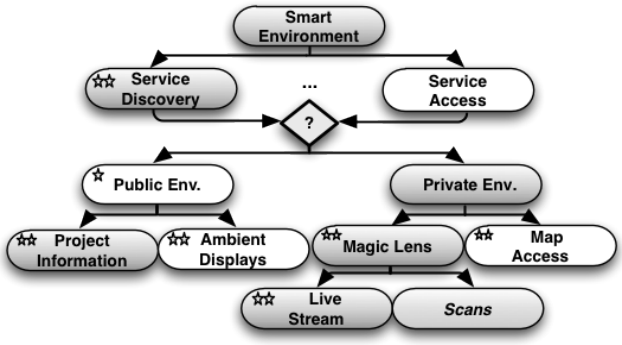


Figure 1. An excerpt of an example pattern language showing ratings as stars, relations as arrows, a necessary decision denoted by the question mark box, pattern sequences (gray) and one anti pattern node (*Scans*).

 ISBN: 978-1-61208-158-8

smaller group of authors and their, sincerely founded, domain knowledge. Additionally, writers workshops that help to formulate and refine patterns are organized during special Pattern Language of Patterns (PLoP) conferences [27]. Our approach addresses a larger amount of users especially application designers using the patterns and researchers familiar with the domain. The community will be given ways to comment on existing patterns and give suggestions for their refinement, provide additional references in literature or name counter-examples that confute the message of a design pattern.

Users can also propose new patterns to be linked within the pattern language structure. This way, the *vividness* of the pattern language is to be increased making it adoptable or extensible for more application domains within the UbiComp scope. Of course, rules and processes need to be defined in future work to guarantee controlled contributions.

Discussion and Refinement: Due to the fact that the patterns should not only be provided by one small group of authors and be ranked by them, rules and processes to propose new patterns or to discuss existing ones are needed.

This may result in a change of ranking, references to public work or parts of the description of the pattern. In case that only a smaller set of authors recommends and formulates patterns, the community feedback can also be interesting in order to see what concepts where really working and what needs to be changed in another iteration so that a pattern refinement is achieved.

Notes on the pattern hierarchy

The pattern hierarchy takes the dimension of time and implementation detail into account; the further the reader follows the structure, the more she learns about underlying concepts and design possibilities. At the lower levels of patterns, suggestions for implementations will follow.

B. Proposed Pattern Structure

According to the already existing pattern structures we describe the intended fields for an *UbiComp application design pattern* in the following. Some properties that were already defined by [9] and [10] were changed compared with the existing ones, new fields are marked by an asterisk(*).

Name: A pattern's name fulfills a vocabulary function by summarizing the intention of the pattern and the core of its solution. Finding a descriptive name which covers the pattern's intention and problem scope is not an easy task (cf. [10], p.65).

Ranking*: In the approaches described so far, patterns were ranked based on the experience of the author group providing the patterns eventually influenced by first reader groups. The ranking should be extended in such a way that readers and implementers of the pattern are enabled to provide feedback about the quality of the pattern. On the one hand with regards to readability and understandability, on the other hand regarding applicability, i.e., design success. That way, patterns can be refined or changed in an iterative way - also by third-party readers and consumers, respectively.

Illustration: Depending on the problem context, the media used for providing a quick glance at the problem should sensitize the reader and inform her quickly about the general design problem that is addressed by the pattern. The illustration may contribute to decision-making whether the pattern is relevant to the current situation.

Context(*): normally, the context is clearly defined by the pattern language structure. However, due to the restriction of design space given by decision nodes, readers have to keep track of taken decisions on the path to the particular pattern. It is possible that the problem domain is narrowed or changed to different problem cases by taking different decisions. This is only the case if nodes may have multiple parents.

Problem and Forces: In order to really capture the problem the current pattern is intended to solve, Borchers suggests to describe it as conflicting interests in a situation (opposing forces) that may result from physical facts but also may include social, economical and psychological constraints [9], [10]. These forces limit the design space or can even be contradictory such that they need to be discussed and eventually a compromise for a solution needs to be found. An example for a physical constraint is given by Alexander; Normally, a table in the center of a room pulls the visitors to that place. An opposing force is that people also like to stand close to large windows [9].

Example (Scenario): The inductive approach of providing examples helps novice readers to better understand the problem and solution described in a pattern. Secondly, experienced readers and professionals can already derive verifiable evidence for a working design solution. The example should be easy to understand and only cover the basic idea behind the pattern in order to provide a quick orientation to the readers.

Solution / Reason(*): This part provides a generally applicable design solution based on the lessons learned from the examples and literature references. It summarizes the central message of the design pattern telling the reader how to handle a specific design problem. In case that the pattern provide an anti-solution, this part turns into a description of the reasons for the failure of the concept. Both, the solution or the reasons should be described succinctly and therefore deliver the key message of the design pattern.

Diagram(optional): Depending on the current problem, an appropriate medium is to be chosen to illustrate and summarize the design solution. The diagram also servers as reading and understanding help and thus provides keeping an overview of the pattern during a quick scan of the pattern language. Additionally, the diagram is intended to assist in remembering the design solution.

Evidence in Literature*: Besides examples that already provide some evidence for a working solution, we regard

 ISBN: 978-1-61208-158-8

references in literature as very important for supporting the design recommendation inside a pattern. That way, application designers and researchers have another way of judging a design pattern besides the collaborative ranking mechanism described above. They can strengthen or weaken the design patterns statement by a custom literature research or custom evaluations that were published. Welie presents a similar concepts on his website where links to literature and a section for discussions are available [28].

Currently, design knowledge can be taken from literature and existing design pattern languages or best practices. In the large scope of UbiComp application design, the gap between knowledge in literature and (not yet) existing design patterns could be bridged.

As time passes, it could even happen that a pattern for design recommendation can be turned into an anti-pattern and vice versa, depending on evaluations, ranking and findings in application and research.

References*: These are pointers to other patterns of interest. In the pattern language approaches so far, the reader was encouraged to have a look at the referred patterns. The successors were mostly more detailed descriptions of a design concept based on size [9], time [10] or technical implementation level [12]. We extend the concept in three ways:

- By introducing *decisions*, only a subset of followers can be chosen by the readers. After choosing a subset of linked patterns, other successors may no longer be of interest for the pattern sequence.
- Additionally, the reader is supported by *most-read successors* based on the reading choice of others. This way, well-proven pattern combinations and sequences can be defined.
- It also has to be checked whether a certain reference is still valid. Earlier decisions may have cancelled out certain references. This is a big difference compared to the other pattern language approaches. Here, the hierarchy was mainly used for the grade of detail.

Decision-attribute*: If set, this new feature indicates a choice that needs to be made by the reader for the references leading to the current patterns successors. In order to capture a wide application domain, the pattern language needs ways to differentiate between alternatives. They concentrate on certain aspects of application design contexts, e.g., private vs. public information display or personal vs. shared devices.

Depending on the choice, a subset of following design patterns is available; others are cancelled out for that design property.

Anti-Pattern-attribute*: The intended pattern language approach also incorporates the explicit formulation of anti-patterns besides design patterns.

Marked as an anti-pattern the described design approach should be read as a DON'T rule for design. It is important that not every failed approach is documented by an anti-pattern. Thus, only surprising, non-trivial results should be documented supporting application designers that might have a similar idea to avoid the same design flaw.

Additionally, anti-patterns can also provide ideas for alternative concepts; once the application designer has followed a pattern sequence, she could also think of changing some parameters. An already formulated anti-pattern could help to avoid an already disproven idea and to try out yet another concept.

Counter-attribute*: The counter attribute is intended to keep track of pattern combinations indicating the strength of relation between the current pattern and its predecessors and its successors, respectively. The more frequent a pattern combination is selected by readers, the stronger the relation between them could be.

Accompanied by the ranking of each pattern, statements about pattern combinations and pattern sequences could be made.

VI. Summary

In this work, we addressed the requirements for establishing a pattern language structure for UbiComp application design as proposed in [2]. Based on existing concept from architectural, HCI and HCHI design patterns formulated by [9], [10], [12], we formulated new features for the existing pattern language structures.

The intention is to widen the application scope and to involve the community reading and working with given patterns and providing ways to express the feedback by rating and refining existing patterns or contributing new findings as patterns to be integrated into the pattern language.

In order to integrate patterns into the structure, the traditional pattern structure was extended by new features or existing features were partially refined to fit into the scope of the intended pattern language approach.

The extended pattern language concepts and pattern structure will result in a collaborative community platform, called *Pattern Management System (PMS)*, incorporating rules and processes for changing and refining existing patterns as well as ways to contribute new application design knowledge as patterns.

VII. Future Work

Future work will cover the following steps that build on top of each other:

Firstly, an initial set of UbiComp application patterns will be formulated and arranged in the pattern language.

Secondly, a first version of an online community portal to read and rate the existing patterns will be deployed.

Thirdly, rules and processes for rating, discussing, refining and contributing to the pattern language will be formulated and implemented. The intention is to support pattern authors willing to extend or modify the existing structure. The

 ISBN: 978-1-61208-158-8

mechanisms for the pattern discussion and refinement will be based on the concepts applied during writers workshop at Pattern-Language of Patterns (PLoP) conferences [27].

Currently, we also explore ways to implement the pattern and pattern language structure based on ontologies.

ACKNOWLEDGMENT

Irina Astrovas work was supported by the Estonian Centre of Excellence in Computer Science (EXCS) funded mainly by the European Regional Development Fund (ERDF). René Reiners' work was supported by the European Commission within the FP7-SECURITY project BRIDGE (no. 261817).

REFERENCES

[1] M. Weiser, "The computer for the 21st century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.

[2] R. Reiners, "Towards a Common Pattern Language for Ubicomp Application Design - A Classification Scheme for Ubiquitous Computing Environments -," in *Proceedings of the International Conferences on Pervasive Patterns and Applications (PATTERNS 2010)*, IARIA Conference. City, Portugal: Think Mind(TM) Digital Library, Nov. 2010, pp. 28–33.

[3] R. Ballagas, S. Kratz, and E. Yu, "REXplorer: A Mobile, Pervasive Spell- Casting Game for Tourists," *Architecture*, pp. 1–6, 2007.

[4] N. Henze, R. Reiners, X. Righetti, E. Rukzio, and S. Boll, "Services surround you," *The Visual Computer*, vol. 24, no. 7-9, pp. 847–855, Jul. 2008.

[5] R. Wetzel and I. Lindt, "The Magic Lens Box: Simplifying the Development of Mixed Reality Games," pp. 479–486, 2008.

[6] J. Schöning, M. Rohs, and S. Kratz, "Map Torchlight: A Mobile Augmented Reality Camera Projector Unit," *Information Systems*, 2009.

[7] R. Reiners, M. Jentsch, and C. Prause, "Interaction Metaphors for the Exploration of Ubiquitous Environments," in *Scenario*. Brussels, Belgium: International Conference "ICT that Makes the Difference", 2009.

[8] R. Reiners and V. N. Wibowo, "Prototyping an Extended Magic Lens Interface for Discovering Smart Objects in a Ubiquitous Environment," in *Proceedings of the IADIS International Conference Applied Computing 2009*, W. Hans and P. T. Isaías, Eds., IADIS. Rome: IADIS Press, Nov. 2009.

[9] C. Alexander, *A Pattern Language: Towns, Buildings, Construction*. New York, New York, USA: Oxford University Press, 1977.

[10] J. Borchers, *A Pattern Approach to Interaction Design*, 1st ed. John Wiley & Sons, 2001.

[11] J. Tidwell, *Designing Interfaces*, 1st ed. O'Reilly Media, 2005.

[12] T. Schümmer and S. Lukosch, *Patterns for Computer-Mediated Interaction*. Chistester, West Sussex, England: John Wiley & Sons, 2007.

[13] E. Gamma, R. Helm, R. E. Johnson, and J. Vlissides, *Design Patterns. Elements of Reusable Object-Oriented Software*, 1st ed. Amsterdam: Addison-Wesley Longman, 1995.

[14] S. Montero, P. Díaz, and I. Aedo, *A Semantic Representation for Domain-Specific Patterns*. Springer-Verlag, 2005, pp. 129–140.

[15] J.-M. Rosengard and M. F. Ursu, *Ontological Representations of Software Patterns*. Springer Berlin / Heidelberg, 2004, vol. 3215, pp. 31–37.

[16] D.-K. Kim, R. France, S. Ghosh, and E. Song, "A UML-Based Metamodeling Language to Specify Design Patterns," in *Proceedings of the Workshop Software Model Eng. (WiSME) with Unified Modeling Language Conf. 2003*, 2003.

[17] G. Sunyé, A. L. Guennec, and J.-M. Jézéquel, "Design Pattern Application in UML," in *Proceedings of the 14th European Conference on Object-Oriented Programming (ECOOP '00)*. London, UK: Springer-Verlag}, 2000, pp. 44–62.

[18] M. Fontoura and C. Lucena, "Extending UML to Improve the Representation of Design Patterns," *Journal of Object-Oriented Programming*, vol. 13, pp. 12–19, 2001.

[19] (2011, July). [Online]. Available: http://uml.org

[20] L. Pavlič, M. Heričko, V. Podgorelec, and I. Rozman, "Improving Design Pattern Adoption with an Ontology-Based Repository," *Informatica*, vol. 33, pp. 189–197, 2009.

[21] A. Cornils and G. Hedin, "Tool support for design patterns based on reference attribute grammars," in *Proceedings of WAGA'00*, Ponte de Lima, Portugal, 2000.

[22] A. H. Eden, A. Yehudai, and J. Gil, "Precise Specification and Automatic Application of Design Patterns," in *12th IEEE International Conference on Automated Software Engineering (ASE'97) (formerly: KBSE)*, 1997.

[23] J. M. Smith, D. Stotts, and C. Hill, "Elemental Design Patterns : A Link Between Architecture and Object Semantics Elemental Design Patterns - A Link Between Architecture and Object Semantics," 2002.

[24] T. Taibi, "Formal Specification of Design Patterns - A Balanced Approach," *Journal of Object Technology*, vol. 2, no. 4, pp. 127–140, 2003.

[25] (2011, July). [Online]. Available: http://semanticweb.org/wiki/Main_Page

[26] R. Reiners, "An Extended Pattern Language Approach for UbiComp Application Design," in *Bonner Informatiktage*, 2011.

[27] (2011, July). [Online]. Available: http://www.hillside.net

[28] (2011, July). [Online]. Available: http://www.welie.com/patterns/

 ISBN: 978-1-61208-158-8

Word Spotting for Arabic Handwritten Historical Document Retrieval using Generalized Hough Transform

Nabil Aouadi
UTIC: research Unit of Technologies of Information and Communication
ESSTT: High School of Sciences and Techniques of Tunis, University of Tunis, Tunisia
nabil.aouadi@utic.rnu.tn

Afef Kacem
UTIC: research Unit of Technologies of Information and Communication
ESSTT: High School of Sciences and Techniques of Tunis, University of Tunis, Tunisia
afef.kacem@esstt.rnu.tn

Abstract— **Because of the high noise levels in historical documents and the great amount of variability in handwriting, handwritten historical documents are currently transcribed by hand. Easy access to such documents requires an index, which is currently created manually at great cost. The goal of the Word Spotting idea, applied to handwritten documents, is to greatly reduce the amount of annotation work that has to be performed, by grouping all words into clusters. This paper explores the use of GHT (Generalized Hough Transform) in case of word spotting for Arabic handwritten historical document retrieval. We applied GHT to identify all positions of a given word in a document. It has the advantage of being relatively unaffected by image noise. Experiments that have been conducted on the historical documents of the Tunisian national Archive show the advantage of the proposed approach.**

Keywords-Generalized Hough Transform; word spotting; pattern recognition; image processing

I. INTRODUCTION

As historical library collections across the world hold huge numbers of handwritten documents, digitizing these manuscripts should be of great interest since their content can be conserved and made available to a large community via the Internet or other electronic media. Such corpora can nowadays be shared relatively easily, but they are often large, unstructured, and only available in image formats, which makes them difficult to access. In particular, finding specific locations of interest in a handwritten image collection is generally very tedious. Easy access to such collections requires an index, which is currently created manually at great cost in terms of time and money.

Due to the large number of handwritten manuscripts, the great amount of variability in handwriting, the high noise levels in historical documents, the transition from traditional to digital libraries pose a great challenge. As automatic handwriting recognizers fail on historical manuscripts, the word spotting technique has been developed: the words in a collection are matched as images and grouped into clusters which contain all instances of the same word. By annotating "interesting" clusters, an index that links words to the locations where they occur can be built automatically. Due to the noise in historical documents, selecting the right features for matching words is crucial. This paper explores the use of GHT [18] technique to identify all positions of a given word in a document.

The HT (Hough Transform) is a technique which can be used to isolate features of a particular shape within an image [17]. Because it requires that the desired features be specified in some parametric form, the *classical* Hough Transform is most commonly used for the detection of regular curves such as lines, circles, ellipses, etc. A *generalized* Hough Transform can be employed in applications where a simple analytic description of a feature(s) is not possible.

The idea is to detect words through their parts of words outlines using GHT. This technique is also known to be tolerant of gaps in feature boundary descriptions and relatively unaffected by image noise. The outline of the paper is as follows. Section II presents some works devoted to the word spotting for Latin handwritten documents. Section III highlights on characteristics of the old Arabic handwriting and the difficulties which it poses. Section IV gives some details of the proposed system of word spotting for Arabic handwritten historical document retrieval based on GHT technique. Section V draws some conclusions and gives an outlook for future work

II. STATE OF ART

Word Spotting is an approach that clusters word images to identify and annotate content-bearing words in a collection. The goal is to greatly reduce the amount of annotation work that has to be performed, by grouping all words into clusters. Ideally, each cluster contains words with the same annotation. Once such a clustering of the data set exists, the number of words contained in a cluster can be used as a cue for determining the importance of the word as a query term. For example, highly frequent terms, such as *the*, *of*, etc. are *stop* words and can be discarded. All clusters with terms that are deemed important can then be manually annotated. This makes it possible to construct a partial index for the analyzed document collection, which can be used for retrieval.

Search works on word spotting for Latin handwriting attracted much attention [1][2][3] and have offered a starting point for the development of new systems. Several methods have been compared in [4] and showed that the

 ISBN: 978-1-61208-158-8

best accuracy was obtained with the method based on DTW (Dynamic Time Warping): an algorithm for measuring similarity between two sequences which may vary in time or speed [19].

Word spotting in the Arabic script dates back several years when we unfortunately offered little work and solutions for handwritten Arabic script. In [5], the study discusses the performance achieved by applying the concept of Word Shape applied on Arabic Handwritten Documents.

The use of CEDEARABIC system becomes very frequent. The word spotting in this system is divided into two stages: indexing and search. Each image generates a set of queries by sub-images. Note that most of the functionality of the system CEDEARABIC, including word spotting, are those of the American Fox [6][7].

You et al. [8] presented a hierarchical Chamfer matching scheme as an extension to traditional approaches of detecting edge points, and managed to detect interesting points dynamically. They created a pyramid through a dynamic thresholding scheme to find the best match for points of interest. The same hierarchical approach was used by Borgefors [9] to match edges by minimizing a generalized distance between them.

Rothfeder et al. [10], Srihari et al. [11] presented a system for spotting words in scanned document images for three scripts: Devanagari, Arabic, and Latin. Their system retrieved the candidate words from the documents and ranked them based on global word shape features.

An algorithm for robust machine recognition of keywords embedded in a poorly printed document was presented by Kuo and Agazzi [12]. For each keyword, two statistical models were generated – one represents the actual keyword and the other represents all irrelevant words. They adopted dynamic programming to enable elastic matching using the two models.

A language independent system for preprocessing and word spotting of historical document images was presented by Moghaddam et al. [13], which has no need for line and word segmentation. In this system, spotting is performed using the Euclidean distance measure enhanced by rotation and DTW.

As far as we know, up till now, there is no system that allows word spotting for handwritten documents using GHT which seems to be of great interest to detect words.

III. CHARCTERISTICS OF ARABIC HIRISTOCAL HANDWRITING

A summary of the features of Arabic writing appears below.

- Arabic text, both handwritten and printed, is cursive. The letters are joined together along a writing line (see Figure 1). This is similar to Latin 'joinedup' handwriting, which is also cursive, but in which the characters are easier to separate.
- In contrast to Latin text, Arabic is written right to left, rather than left to right.
- Arabic contains dots and other small marks that can change the meaning of a word, and need to be taken into account by any computerized recognition system.
- The shapes of the letters differ depending on whereabouts in the word they are found. The same letter at the beginning and end of a word can have a completely different appearance as shown in Figure 2. Along with the dots and other marks representing vowels, this makes the effective size of the alphabet about 160 characters.

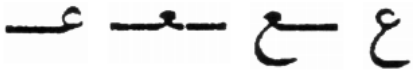


Figure 1. Different forms of the Arabic letter "ع". From left to right beginning, middle, end, isolated

- In Arabic handwriting the word is composed of one or several of parts of word. A PAW (Part of Arabic Word) is a connected component which can refer to a diacritic sign, a single letter, a sequence of letters or whole word (see Figure 2).

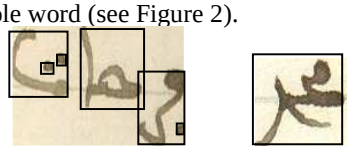


Figure 2. Example of a words composed of several number of PAWs

- Because letters can be horizontally and/or vertically ligaturated, as shown in Figure 3, the segmentation is not an easy busness.

Figure 3. Problem of segmentation due to letter ligature

For Arabic historical handwriting, other difficulties are compounded. In the historical documents that we handle, we find that for some letters, the number and/or position of their diacritic points were changed. For example, the letter ڤ is written ق: a single diacritic point above the letter body instead of two whereas the letter ڤ is written with a single diacritic point below the letter body: ڤ.

IV. PROPOSED SYSTEM

The proposed system, described here, has been built on a subset of the Tunisian National Archive collection. We are interested by documents that are from the 19th century and correspond to enumeration registers at Tunisian protectoral period. Figure 4 displays a small portion of this manuscript: a set of lines composed of list of names. These documents are written in one author's hand. This reduces the amount of handwriting variations that have to be compensated for.

The proposed system retrieves actual handwritten pages (not transcriptions) given text queries. It is based, as it will

 ISBN: 978-1-61208-158-8

be explained, GHT technique that we have improved by a clustering process. The system consists of training and recognition phases.

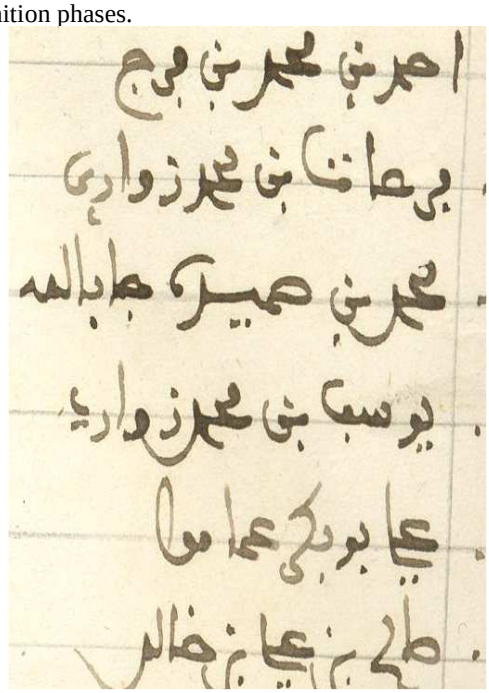


Figure 4. Example of page of the enumeration register

At the training phase, the system generates patterns of PAWs considering different samples. The PAW pattern is created from its contour points and gravitational center coordinates. PAW patterns are then registred in a table of reference as it will be explained later. Afterwards, the system builds a dictionary which includes word queries. For each word, the system saves its composing PAWs, the corresponding *d-cluster* (the minimal distance between word clusters which is empirically determined and for each PAW, the associated pattern, the Hough threshold (number of minimal votes for the PAW) and the Hough threshold for points that are close to the voting points. In phase of recognition, the system, given the word to spot, it computes GHT parameters for each of PAWs and scans the page pixel by pixel looking for the voting points and their neighborhood. In fact, the PAW patterns will largely vote for similar patterns and will form clouds of points, also called voting clusters, in the Hough space. If the distance between voting clusters is less than *d-cluster* then the system merge them into one cluster. This process has significantly improved the performance of word spotting as it will be demonstrated later. Let us recall the GHT principle then consider its application for word spotting.

A. GHT Technique

Hough Transform is a feature extraction technique to find imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm for computing the Hough Transform.

The classical HT was concerned with the identification of lines in the image, but later the Hough Transform has been extended to identifying positions of arbitrary shapes, most commonly circles or ellipses. The GHT, introduced by D.H. Ballard in 1981 [18], is the modification of the HT using the principle of template matching. This modification enables HT to be used for not only the detection of an object described with an analytic equation (e.g. line, circle, etc.). Instead, it can also be used to detect an arbitrary object described with its model.

The problem of finding the object (described with a model) in an image can be solved by finding the model's position in the image. With the GHT, the problem of finding the model's position is transformed to a problem of finding the transformation's parameter that maps the model into the image. As long as we know the value of the transformation's parameter, the position of the model in the image can be determined.

B. Word spotting using GHT Technique

The original implementation of the GHT uses edge information to define a mapping from orientation of an edge point to a reference point of the shape. In the case of a binary image where pixels can be either black or white, every black pixel of the image can be a black pixel of the desired pattern thus creating a locus of reference points in the Hough Space. Every pixel of the image votes for its corresponding reference points. The maximum points of the Hough Space indicate possible reference points of the pattern in the image.

A table of reference, called R-Table, defines the correspondence between the space image and parametric space [13]. The GHT acts on characteristic points of the image generally contour image. It makes it possible to describe the form by R-table in phase of training, while, in phase of recognition it exploits various R-tables obtained in the phase of training to generate spaces of votes which make it possible to carry out classification [14].

To build R-tables, our system extracts all the connected components of words which are the set of PAWs and generates a pattern for each one of them. The pattern is deduced from PAW contour points and a reference point, here the PAW gravitational center *G*. For each point *P*, as shown in Figure 5, the system selects the two neighboring points P_1 and P_2. Then, it determines the tangent line at *P* which is parallel to the line (P_1P_2) passing through *P*. Let β be the angle between the tangent line at *P* and the horizontal (x-axis). Let α be is the angle between the orthogonal line to the tangent at *P* that passes through *P* and the line (*GP*). All computed values are stocked in R-table in function of β as displayed in Table 1.

 ISBN: 978-1-61208-158-8

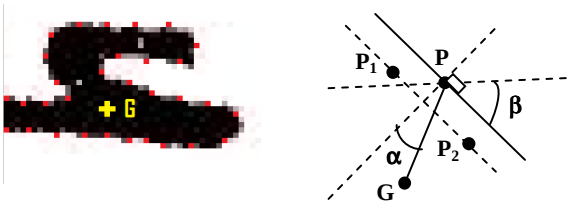


Figure 5. PAW pattern creation

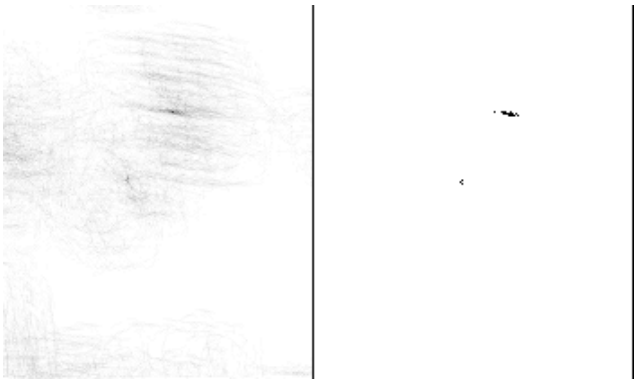
TABLE 1. R-TABLE CONSTRUCTION

β_j	(α_i, P_iG)
30°	(α_i, P_1G), (α_i, P_2G), (α_i, P_3G), …
40°	…
…	….

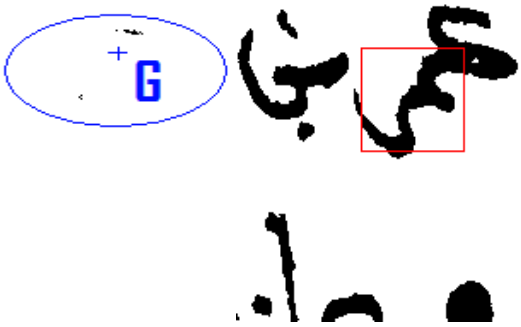
During the recognition phase, here for word spotting, the system extracts the contour of PAWs and passes a sliding window through the contour. At each passage and for a point inside the window, the system determines the tangent line at this point with the ends of the contour points that touch the border of the window. Then the system calculates β and consults R-table in search of couples α and P_iG which coincide with β. The found couples are the voting points. Finally, if the number of voting points is above a Hough threshold then the PAW is located. As our objective is to locate the entire word in the document image, the same above process is repeated for the rest parts of word. We also considered the voting clusters of PAWs and the distance, *d-cluser*, between them.

Let us give an example of word spotting using GHT for the word "عمر". In Figure 6 (a), many clusters of voting points are formed for the word parts: "ع" and "مر". The voting cluster is simply a transcription of the letter or word part image where are recorded the votes of each pixel. In Figure 6 (b), the system selects points that have the most votes. When the voting clusters are close in term of Euclidean distance: *d-cluster*, as shown in Figure 6 (c), they will be merged into one cluster which represents the votes of individual patterns that compose the word to be located.

(a) GHT application on word parts : "ع" and "مر"



(b) Selection of points that have the most votes



(c) Combinations of neighboring clusters by *d-cluster* distance

Figure 6. Word spotting of the name "عمر" using GHT

C. System evaluation

To evaluate performance of our system, we used a database which consists of 23 pages. Each page includes about 36 lines. Each line contains 4 names on average. So we handle a total of 3312 words. These pages are scanned with a resolution of 300 dpi and saved in TIFF format.

Figure 7 displays the results of a part of the word name "علي" retrieval. Several tests were carried out on other words. The observed results are quite satisfactory. Experiments that have been conducted on the historical documents of the Tunisian national Archive show the advantage of the proposed approach. This approach is relatively unaffected by image noise. But the main drawbacks of this approach, based on GHT, are its substantial computational and storage requirements that become acute when word orientation and scale have to be considered. At the current state of the system, the computing time is about one second.

 ISBN: 978-1-61208-158-8

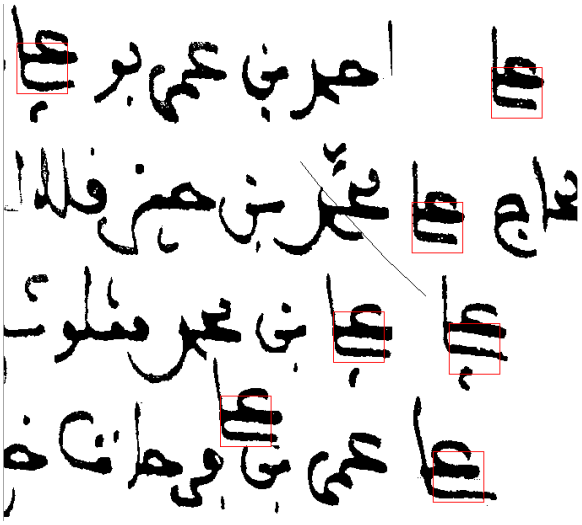


Figure 7. Word spotting of the name "علي" using GHT

V. CONCLUSION AND FUTURE WORK

This paper discusses and summarizes the work covered, its contribution to the word spotting for Tunisian historical handwritten documents. The proposed system is based on GHT to identify all positions of a given word in a document. The GHT is generally used to detect arbitrary shapes. As handritten words have no simple analytical forms, we tried here to benefit from this technique for word spotting. The system attempts to improve performance by grouping vote clusters of letters or part of words which compose the enquires words. How to choose models to optimize GHT? How to enhance system robustness against noise and poor quality of pages? There is still a lot to do but it is really worthwhile.

ACKNOWLEDGEMENT

We thank O. Elghoul for kindly providing us with the GHT source code.

REFERENCES

[1] N. Ben Amara, A. Belaïd, and N. Ellouze, "Modélisation Pseudo bidimensionnelle pour la Reconnaissance de Chaînes de Caractères Arabes Imprimés," Proc. CIFED 98, Québec, Canada, pp. 131-140, 1998.

[2] S. Kuo and O. Agazzi, "Keyword Spotting in Poorly Printed Documents using 2-d Hidden Markov Models", IEEE Trans. PAMI, 16, pp. 842–848, 1994.

[3] M. Burl and P. Perona, "Using Hierarchical Shape Models to Spot Keywords in Cursive handwriting," IEEECS Conference on Computer Vision and Pattern Recognition, pp. 535–540, June 1998.

[4] A. Kolz, J. Alspector, M. Augusteijn, R. Carlson, and G. V. Popescu, "A Line-oriented Approach to Word Spotting in Handwritten Documents," Pattern Analysis and Applications, 2(3), pp. 153–168, 2000.

[5] R. Manmatha and T. M. Rath, "Indexing of Handwritten Historical Documents-recent Progress," Proc. SDIUT 03, pp. 77–85, 2003.

[6] B. Zhang S. N. Srihari and C. Huang, "Word Image Retrieval using Binary Features," Proc. DRR 04, SPIE Vol. 5296, pp. 45–53, 2004.

[7] S. N. Srihari, S.H. Cha, H. Arora, and S. Lee, "Individuality of handwriting," Forensic Sciences, 47(4), pp. 856–872, 2002.

[8] S. N. Srihari, B. Zhang, C. Tomai, S. Lee, Z. Shi, and Y. C. Shin, "A system for handwriting matching and recognition," Proc. SDIUT 03, Greenbelt, MD, pp. 67–75, 2003.

[9] J. You, E. Pissaloux, W. Zhu, and H. Cohen, "Efficient Image Matching: A hierarchical Chamfer Matching Scheme via Distributed System", Real-Time Imaging, 1(4), pp. 245 – 259, 1995.

[10] G. Borgefors, "Hierarchical Chamfer Matching: Aparametric Edge Matching Algorithm," IEEE Trans. Pattern Anal. Mach. Intell., 10(6), pp. 849–865, 1988.

[11] J. L. Rothfeder, Shaolei Feng, and Toni M. Rath, "Using Corner Feature Correspondences to Rankword Images by Similarity," Proc. DIAR 03, Madison, WI, June 2003.

[12] S. Srihari, H. Srinivasan, C. Huang, and S. Shetty, "Spotting Words in Latin, Devanagari and Arabic scripts," Vivek: Indian Journal of Artificial Intelligence, 16(3), pp. 2–9, 2003.

[13] S. Kuo and O. Agazzi, "Keyword Spotting in Poorlyprinted Documents using Pseudo 2-d Hidden Markov Models," IEEE Trans. Pattern Anal. Mach. Intell., 16(8), pp. 842–848, 1994.

[14] R. Moghaddam, D. Rivest-Hénault, and M. Cheriet, "Restoration and Segmentation of Highly Degraded Characters using a Shape-Independent Level Set Approach and Multi-level Classifiers," Proc. ICDAR 09, Barcelona, Spain, pp. 828–832, 2009.

[15] M. Ulrich, C. steger, A. Baumgartner, and H. Ubner, "3 Real Time Object Recognition Using a Modified Generalized Hough Transform," Eckhardt Seyfert, editor, photogrammetrie -- Fernerkundung-- Géoinformation : Geodaten schaffen verbindungun, 21. Wissems chaftlich-Technische Jahrestagung der DGPF, Berlin, 2001, pp 571-578.

[16] S.Touj, N. Ben Amara, and H. Amiri, "Reconnaissance de l'écriture Arabe Imprimée par une Approche de Segmentation par Reconnaissance Basée sur la Transformée de Hough Généralisée," SETIT 03, Mars, Sousse, Tunisie 2003.

[17] R. Fisher, S. Perkins, A. Walker, and Erik Wolf, "Image Transforms - Hough Transform," Homepages.inf.ed.ac.uk. Retrieved 2009-08-17.

[18] D.H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes", Pattern Recognition, Vol.13, No.2, pp. 111-122, 1981.

[19] C. S. Myers and L. R. Rabiner, "A Comparative Study of Several Dynamic Time-warping Algorithms for Connected Word Recognition," The Bell System Technical Journal, 60(7), pp. 1389-1409, September 1981.

 ISBN: 978-1-61208-158-8

Practical Math and Simulation in Software Design

Jerry Overton
Computer Sciences Corporation (CSC)
joverton@csc.com

Abstract – Formal verification of a software design is often much more costly than producing the design itself; and formal methods usually have limited use in real-world software design. In this work, we propose cost-effective methods – based on software design patterns – of applying mathematics and simulation to real-world software designs.

Keywords – Software Design Pattern, Practical Formal Method, POAD Theory

I. INTRODUCTION

The cost of using formal methods to verify a software design is usually an order of magnitude greater than the cost of creating the design itself [1]. For many projects, formal methods are only worth using for reducing the risk of the most serious errors – flaws that may affect safety, for example [2]. We propose practical (cost-effective) methods of mathematics and simulation for real-world software designs. We introduce a method for validating the adequacy of software designs based on the mathematics of Pattern-Oriented Analysis and Design (POAD) Theory [3], [4] and a technique for simulating software designs based on fuzzy logic. The result is the practical application of formal methods to a real-world software design. That is, we apply to an actual design for working software a formal method for validating the design and automating the analysis; and we do so in less time than it took to create the software design.

We start by specifying the design for a real-world problem of interest. We continue by using POAD Theory to structure an adequacy argument for the design. We apply a simulation technique based on fuzzy logic to fully justify our adequacy argument. Finally, we close with an analysis of our technique and conclusions about the significance of this research.

II. STATE OF THE ART

Previous works like [5], [6] and [7] propose methods of pattern-based reasoning using rules generalized from specific design experiences. The works of [5] and [6] build rules from observed correlations between patterns and a particular software quality. The work of [7] places all possible pattern implementations into a limited set of categories, and then derives rules for each category. Works like [5], [6] and [7] required time-consuming tailoring of general rules before predictions could be made about real-world software designs. For example, the method described in [7] required users to make a complete formal model of a working system before it could be used to make predictions about the system's design. By contrast, we make predictions about software designs based on subjective arguments and use general rules only to structure those arguments and demonstrate validity. Our method allows us to make predictions about the consequences of a design with an effort much smaller than that required by previous works.

III. A COLLABORATIVE SYSTEM DESIGN

Figure 1 is an example of a collaborative system [8] designed to report the environmental condition of a given region. Each sensor is capable of recording and reporting its local conditions, but to record and report the condition of the entire region requires that all sensor stations cooperate.

The nodes in the network are autonomous and spatially distributed across the region shown. Each sensor is capable of autonomously recording and reporting its local environmental conditions to a controller in its region. Task execution is distributed across multiple nodes since reporting conditions for the entire region requires that a controller communicate with multiple sensor stations. Sensors

 ISBN: 978-1-61208-158-8

can enter and leave the network at anytime. Every station is wirelessly connected to every other station, so no single sensor failure can disrupt the overall network connectivity.

We expect that node failures will be common and that the wireless communication links will be prone to frequent interruptions. For example, the sensor stations are exposed to adverse weather, they are knocked over and broken easily, and they can be expected to run out of power. If any of these things happen at the right time, a controller in a region may miss a sensor update and become out of touch with the current conditions in the region.

Figure 1: Example Collaborative System [9]

A robust design will allow the sensor stations (known as nodes) to both detect and mitigate these kinds of failures. A satisfactory design must satisfy the following requirements.

Req. 1. *Group Communication*. Each node must be able to communicate with all other nodes and detect when a node becomes unresponsive.

Req. 2. *Fault Tolerance*. The network must be capable of using node redundancy to compensate for the loss of any particular node.

Req. 3. *Degraded Mode Operation*. Each node must be capable of performing limited functions while disconnected from the network, and be capable of resuming full function when network communication is restored.

Figure 2 shows our design for a robust collaborative system. We consider Figure 2 a real-world design since it was taken from the design of an actual software system built to provide fault tolerance in collaborative systems [10]. Each *GroupNode* gets its ability to collaborate through an association with a *CommStrategy* object. The *CommStrategy* has an association back to its *GroupNode* in case the *GroupNode* needs to be notified of events from the *CommStrategy*. Using the JGroup communication API [11] the *PushPullStrategy* gives each *PushPullNode* the ability to communicate with other *PushPullNodes*.

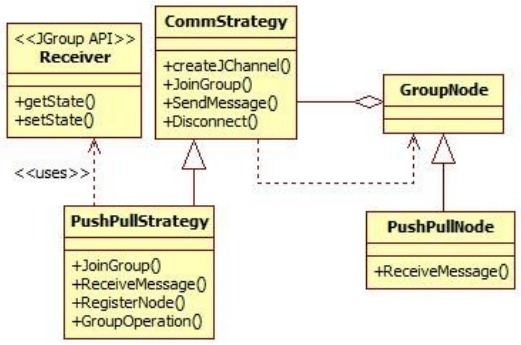


Figure 2: Design for robust collaborative system.

Figure 3 shows how the design works. The Controller relies on either sensor A or sensor B to report temperature for a given region. When the Controller wants a temperature reading from the zone, it joins the zone's group and executes the *commstrategy.GroupOperation()* operation. JGroups elects a leader within the group and calls *getState()* on that node (let's assume that sensor A was chosen). The *getState()* operation of sensor A takes a temperature reading and sets the reading as the operation's return value. JGroups then calls *setState()* on the Controller, passing it the temperature reading from sensor A. In subsequent requests for the zone temperature, if sensor A becomes unresponsive, JGroups will failover to sensor B.

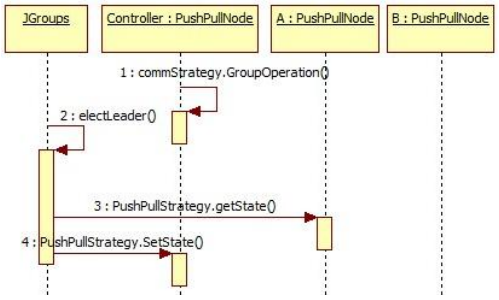


Figure 3: Nodes participating in a group operation

 ISBN: 978-1-61208-158-8

We have a design, but is it adequate: does it solve our problem and satisfy our requirements? In the next section, we use the mathematics of POAD Theory to structure an adequacy argument for the design.

IV. APPLYING PRACTICAL MATHEMATICS

POAD Theory is based on Problem-Oriented Software Engineering (POSE) [12] where engineering design is represented as a series of transformations from complex engineering problems to simpler ones. In POSE a software engineering problem has context (a real-world environment), W; a requirement, R; and a solution (which may or may not be known), S. We write $W, S \vdash R$ to indicate that we intend to find a solution S that, given a context of W, satisfies R. The problem, $CSystem$, of designing a collaborative system can be expressed in POSE as:

$$CSystem: W, S \vdash R$$

(1)

where W is the real-world environment for the system (shown in Figure 1); S is the system itself and R are requirements Req. 1, Req. 2, and Req. 3. Equation (1) says that we can expect to satisfy R when the system S is applied in context W.

POAD Theory uses POSE to represent software design patterns as justification for transforming a complex, unfamiliar problem into simpler, more familiar one [3], [4]. For example, the engineering expertise documented in the Object Group pattern describes how to achieve reliable multicast communication among objects in a network [14]. We can use the engineering judgment in the Object Group pattern (represented as $\ll OG \gg$) to justify a solution interpretation (represented by the rule $[SolInt]$) from $CSystem$ to $Comm$ and Obj. We write this as

$$\frac{Comm\ Obj}{CSystem: W, S \vdash R} [SolInt] \ll OG \gg$$

(2)

Equation (2) implies that if we have a solution to $Comm$ and Obj then we also have a solution to $CSystem$. The reliable multicast communication that we get from the Object Group pattern may be sufficient to satisfy Req. 1 and Req. 2, but we have not yet addressed Req. 3. To satisfy Req. 3, we need to insulate Obj so that it can continue to operate after losing communication with its environment. The Explicit Interface pattern describes how to achieve separation between an object and its environment [15]. We can use the Explicit Interface pattern $\ll EI \gg$ to justify the transformation of Obj into a $Node$ that is separated from its environment by an interface $Intf$.

$$\frac{Comm \dfrac{Intf\ Node}{Obj} [SolInt]}{CSystem: W, S \vdash R} \ll EI \gg [SolInt] \ll OG \gg$$

(3)

Equation (3) is a solution tree with $CSystem$ at the root. Two problem transformations extend the tree upward into the leaves $Comm$, $Intf$, and $Node$. The equation structures an argument whose adequacy is established by the conjunction of all justifications – in this case by the engineering expertise contained in the Object Group pattern $\ll OG \gg$ and the engineering expertise contained in the Explicit Interface pattern $\ll EI \gg$. A solved problem is written with a bar over it; for example, if the Object Group pattern were sufficient to convince us that we have an adequate communication mechanism, then in (3) we could rewrite $Comm$ to appear as $\overline{Comm}$.

The argument represented by (3) is not complete and fully-justified until all leaf problems have been solved. We complete the argument by adding transformations and justifications sufficient to solve $Comm$, $Intf$, and $Node$.

$$\frac{\overline{Recvr}\ [SolInt]}{Comm \ll J_1 \gg} \quad \frac{\overline{PPStrat}\ [SolInt]}{Intf \ll J_2 \gg} \quad \frac{\overline{PPNode}\ [SolInt]}{Node \ll J_3 \gg}$$

(4)

In (4) the problems $Recvr$, $PPStrat$, and $PPNode$ correspond to the *Receiver*, *PushPullStrategy* and *PushPullNode* from Figure 2. The $Recvr$is an implementation of the communication mechanism prescribed by the Object Group pattern, $PPStrat$ is an implementation of the interface prescribed by the Explicit Interface pattern, and $PPNode$ is an implementation of the domain

 ISBN: 978-1-61208-158-8

object prescribed by the Explicit Interface pattern. Although we already have justifications EI and OG; justifications J_1, J_2, and J_3 are assumed, yet unknown. We can consider $CSystem$ solved by finding J_1, J_2, and J_3. In the next section, we use design simulation results to complete our missing justifications.

V. APPLYING PRACTICAL SIMULATION

In this section, we use Fuzzy Inference [16] to simulate the effects of the *Receiver*, *PushPullStrategy* and *PushPullNode* (from Figure 2). Fuzzy inference is based on a generalized modus ponens [16] where arguments take the form:

$$\begin{gathered} If\ A\ Then\ B \\ A' \\ Therefore\ B' \end{gathered} \tag{5}$$

For example, suppose we accepted the general rule that: *if the Object Group pattern were implemented as part of our collaborative system, then the group communication of our system would be good*. If we knew that, in our system, the Object Group pattern were implemented poorly, then fuzzy inference would allow us to conclude that the group communication of the system would also be poor.

In works like [17], [18], and [19] fuzzy logic has been used to reverse engineer design patterns: use fuzzy inference to determine if an existing solution, known to satisfy certain requirements, matches a general design pattern. In this work, we apply that idea in reverse. For a given design pattern, we use fuzzy inference to determine if a particular implementation of that pattern will lead to a solution that we can trust will satisfy particular requirements.

We begin our simulation by creating fuzzy rules [16] that represent the design constraints of (3) and (4):

Rule 1. If the Object Group pattern is implemented then group communication will be good

Rule 2. If the Object Group pattern is not implemented then fault tolerance will be low

Rule 3. If the Explicit Interface pattern is not implemented then degraded-mode operation will not be enabled

Rule 4. If the *PushPullNode* communicates statically then degraded-mode operation will not be enabled

Rule 5. If the *PushPullNode* communicates dynamically and the Explicit Interface pattern is implemented then degraded-mode operation will be enabled

Rule 6. If the *PushPullNode* communicates dynamically and the Object Group pattern is implemented then group communication will be good and fault tolerance will be high.

Each rule makes statements concerning input and output variables. Each variable has membership functions [16] that allow the inference engine to turn the numeric values of the variables into the more intuitive concepts used in the Rules 1-6. For example, we defined membership functions (Poor, Good, and Moderate) for the Group Communication output variable so that a value of 0.7 would be considered mostly moderate, slightly good, and not at all poor. As shown in Figure 4, input variables representing our implementation choices are fed into an inference engine, which has been loaded with Rules 1-6. The inference engine calculates values for fuzzy output variables, which represent the results of the simulation.

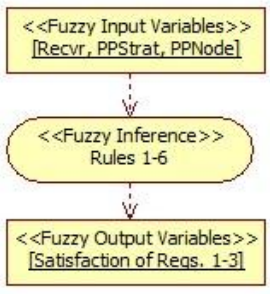


Figure 4: Process flow of the simulation.

We simulate the design choices made in (4) by assigning specific values to the fuzzy input variables $Recvr$, $PPStrat$, and $PPNode$. Our variables range between 0 and 1 and we chose numeric values we believed best reflected our engineering judgment.

 ISBN: 978-1-61208-158-8

We assigned a value of 0.949 to the $Recvr$ variable because JGroups provides a faithful implementation of the Object Group pattern; we assigned a value of 0.762 to the $PPNode$ variable since we consider it a good approximation Object Group pattern's node element; and we assign a value 0.584 to the $PPStrat$ variable because we believe that it is not a very good representation of the intent of the Explicit Interface pattern.

The results of the simulation predict that the design decisions described in (4) and shown in Figure 2 will result in a collaborative system that satisfies Req. 1-3 (see Figure 5). The simulation predicts that the system will have good group communication (0.833), good fault tolerance (0.815), and will operate well in degraded mode (0.807).

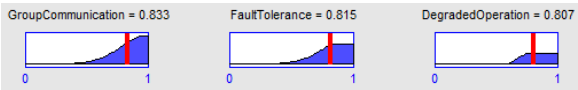


Figure 5: The results of collaborative system simulation.

The positive simulation results along with our analysis of the design spaces provides the justification (J_1, J_2, and J_3), needed to complete the argument (given by (3) and (4)) that the design shown in Figure 2 is adequate.

VI. Analysis and Conclusions

Although the method we introduced does not provide us with proof that our design is adequate, it does provide us with a sound argument for a likely-stable design. Generalized modus ponens – the basis of fuzzy inference – is sound in that its conclusions are true if the premises are true [20]. We can trust the results of our simulation as long as we trust the rules that we establish for governing the simulation. Stable software designs tend to be built from stable sub-designs [21] – although we recognize that using stable sub-designs does not necessarily guarantee overall design stability. Because the mathematics that we use structures arguments based on software design patterns – known-stable designs – we have reason to believe that we are likely arguing for the adequacy of a stable design.

Our method is practical in that, with a relatively small amount of effort, we were able to use math and simulation to discover things about the design that are not obvious. With Eq. 1-5 and the associated explanatory text, we were able to create a mathematical model that had a meaningful correspondence to the collaborative system design in Figure 2. We were able to use those equations to structure an argument for the design's adequacy and to predict that: given the argument structure defined by (3) and (4); and the engineering expertise contained in the Object Group and Explicit Interface patterns; all we needed to validate the design of Figure 2 was to find justifications J_1, J_2, and J_3. We were able to provide that justification using Rule 1-6; fuzzy variable membership function definitions; and fuzzy inference.

Our use of math and simulation is, essentially, an application of analogical reasoning [22] where we draw a comparison between the design of Figure 2 and software design patterns. We were able to argue for the adequacy of our design by replacing the more difficult task of predicting the consequences of the design with the much easier task of comparing the design with known software design patterns. We reason that the closer our design is to the solutions described in the design patterns, the closer our results will be to the consequences described in the design patterns. Our simulation tells us just how close our design needs to be in order to produce satisfying results.

We started with a known software design, structured an argument for the adequacy of the design, and completed the argument using simulation. That order gave us a practical method of validation, but the individual methods are still valid even if we change the order. Suppose, instead, we started by structuring the argument, and then ran the simulation, and last found a software design that fit the argument and simulation. Instead of validating existing software, we would be predicting the existence of unknown software. We would have to simulate software that has not yet been designed; requiring us to guess at a likely implementation. In the future we will investigate if, by changing the order of our method (and overcoming the problems caused by that change), it is possible to use these

 ISBN: 978-1-61208-158-8

same techniques to create an equally practical method of software design prediction.

ACKNOWLEDGEMENTS

We would like to thank Dariusz W. Kaminski of the Marine Scotland directorate of the Scottish Government for his insightful review and commentary.

REFERENCES

[1] D. Jackson. *Lightweight Formal Methods.* FME 2001: Formal Methods for Increasing Software Productivity, Lecture Notes in Computer Science, Volume 2021, pp. 1, 2001

[2] J.P Bowen. *Formal Methods in Safety-Critical Standards*. In Proceedings of 1993 Software Engineering Standards Symposium (SESS'93), Brighton, UK, IEEE Computer Society Press, pp. 168-177, 1993.

[3] J. Overton, J. Hall, L. Rapanotti, and Y. Yu. *Towards a Problem Oriented Engineering Theory of Pattern-Oriented Analysis and Design*. In Proceedings of 3rd IEEE International Workshop on Quality Oriented Reuse of Software (QUORS), 2009.

[4] J. Overton, J. G Hall, and L. Rapanotti. *A Problem-Oriented Theory of Pattern-Oriented Analysis and Design.* 2009, Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, pp. 208-213, 2009.

[5] D. J. Ram, P. J. K. Reddy, and M. S. Rajasree. *An Approach to Estimate Design Attributes of Interacting Patterns*. http://dos.iitm.ac.in/djwebsite/LabPapers/JithendraQAOOSE2003.pdf, Last Accessed: 30 January 2011.

[6] J. Paakki, A. Karhinen, J. Gustafsson, L. Nenonen, and A. Verkamo. *Software metrics by architectural pattern mining*. In Proceedings of the International Conference on Software: Theory and Practice (16th IFIP World Computer Congress), pp. 325–332, 2000.

[7] P. Tonella and G. Antoniol. *Object Oriented Design Pattern Inference.* In Proceedings of the IEEE International Conference on Software Maintenance. IEEE Computer Society Washington, DC, USA, 1999.

[8] T. Clouqueur, K.K. Saluja, and P. Ramanathan. *Fault Tolerance in Collaborative Sensor Networks for Target Detection*. IEEE Transactions on Computers. Vol. 53, No. 3, pp. 320-333, March 2004.

[9] http://www.citysense.net/, Last Accessed: May 2011

[10] J. Overton. *Collaborative Fault Tolerance using JGroups*. Object Computing Inc. Java News Brief, 2007, http://jnb.ociweb.com/jnb/jnbSep2007.html, Last Accessed April, 2011.

[11] The JGroups Project. http://www.jgroups.org/. Last Accessed April, 2011

[12] J. G. Hall, L. Rapanotti, and M. Jackson. Problem-oriented software engineering: solving the package router control problem. IEEE Trans. Software Eng., 2008. doi:10.1109/TSE.2007.70769.

[13] Frank Buschmann, Kevlin Henney, and Douglas C. Schmidt. *Pattern-Oriented Software Architecture: On Patterns and Pattern Languages*, Volume 5. John Wiley & Sons, West Sussex, England, 2007.

[14] S. Maffeis. The Object Group Design Pattern. In Proceedings of the 1996 USENIX Conference on Object-Oriented Technologies, (Toronto, Canada), USENIX, June 1996.

[15] F. Buschmann, K. Henney, and D. Schmidt. *Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing (Wiley Software Patterns Series)*, Volume 4. John Wiley & Sons, 2007.

[16] K. Tanaka. *An Introduction to Fuzzy Logic for Practical Application*. Berlin: Springer, 1996.

[17] J. Niere. *Fuzzy Logic based Interactive Recovery of Software Design*. Proceedings of the 24th International Conference on Software Engineering, Orlando, Florida, USA, 2002, pp. 727-728.

[18] C. De Roover, J. Brichau, and T. D'Hondt. *Combining fuzzy logic and behavioral similarity for non-strict program validation*. In Proc. of the 8th Symp. on Principles and Practice of Declarative Programming, pp. 15–26, 2006.

[19] I. Philippow, D. Streitferdt, M. Riebisch, and S. Naumann. *An approach for reverse engineering of design patterns*. Software Systems Modeling, pp. 55–70, 2005.

[20] S.J. Russell and P, Norvig. *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, NJ: Prentice Hall, 1995.

[21] R. Monson-Haefel. *97 Things Every Software Architect Should Know.* O'Reilly Media, Inc. 2009.

[22] G. Polya. *Mathematics and Plausible Reasoning: Volume II, Patterns of Plausible Inference.* Princeton University Press. 1968.

 ISBN: 978-1-61208-158-8

Evaluating Data Modeling Aspects for Home Telemonitoring

Vilmos Szűcs
Department of Software Engineering
University of Szeged
Hungary
vilo@inf.u-szeged.hu

Ádám Zoltán Végh
Department of Software Engineering
University of Szeged
Hungary
azvegh@inf.u-szeged.hu

Miklós Kasza
Department of Software Engineering
University of Szeged
Hungary
kaszam@inf.u-szeged.hu

Vilmos Bilicki
Department of Software Engineering
University of Szeged
Hungary
bilickiv@inf.u-szeged.hu

***Abstract*—This paper addresses the evaluation of data models designed for Home Telemonitoring to store various data coming from a diverse set of devices on the grounds of two previously developed Telemonitoring Systems. The evaluation is based on quality metrics, measurements, and empirical validation, and it identifies the key differences between a generic and a problem specific data model regarding their main advantages and drawbacks.**

Keywords-telemonitoring; data modeling; data model metrics

I. INTRODUCTION

In recent years, the technical conditions have been changed and the evolution of information technology has made a wide range of affordable devices, almost infinite processing and storage capacities available. Wireless penetration and wireless based communication have also gained significant ground. This trend has led to the development of several embedded and sensor based systems. In the field of the e-Health domain, Telemonitoring Systems started to emerge as an adaptation of these technologies with the objective of providing an efficient basis for home care services [1][2][3][4]. The main goal of these systems is to overcome the typical problems of health care services. On the one hand, they offer a solution for collecting several kinds of physiological data at the patients' home without the need for medical supervision. On the other hand, with the help of data mining and signal processing algorithms they process, sort, and aggregate measured data to transfer and visualize the right information, at the right place, at the right time for medical experts or even for relatives.

To improve the quality and cost effectiveness of health care services, the necessary devices and sensors must be selected carefully to keep the system available for a low price. However, this can lead to dealing with a diverse set of hardware manufacturers and communication protocols. In addition, the various structures of medical data coming from the involved devices must be supported by the system. These issues all affect the design and development of the data model. Furthermore, the data model has an impact on the flexibility, the reusability, and the performance of the developed system. In general, the data modeling process is one of the most critical parts of the design phase in a development process [5].

In the last few years, we have developed two Telemonitoring Systems. The key differences between these Research & Development (R&D) projects were the budget and time constraints, which also affected the requirements and functionalities expected from each system. Regarding these constraints different data models were developed; a generic data model for the long-term project and a problem specific one for the mid-term project in order to keep up with the productivity requirements. The long-term project also provided a generic data visualization, while the mid-term project included accurately defined user interfaces for the doctors. Both systems were evaluated in a Living Lab (LL) experiment with the involvement of real patients and doctors. In this paper, we will examine each data model regarding metrics and measurements, and we will summarize the observations referring to user acceptance and satisfaction relying on the LL tests.

The following section gives a short overview of a typical Telemonitoring System and the requirements for its dataflow and data model. The next part of the document describes data modeling principles and metrics to define what makes a data model good and how we can measure its propriety and quality. The forthcoming section introduces the examined data models. The next part aims to present the comparison and evaluation of the models, which is followed by the analysis of user acceptance. Finally, the paper ends with a summary of the results and a conclusion on the different data modeling aspects presented in this paper.

II. THE ROLE OF DATA MODELS IN TELEMONITORING

Telemonitoring Systems are designed to help with the collection of different physiological parameters of people suffering from various diseases. They offer a remote care solution for medical experts and support the communication between the patients and their care providers while the patients can stay at home. To reach its goals, the system includes a diverse set of active and passive sensors, and a so-called Hub placed in the home of the patient. The communication between the sensors and the Hub is prevalently based on wireless technology [6], mainly over bluetooth or zigbee connections. The Hub is responsible for managing sensors, collecting measured data and transmitting these data to a central data

 ISBN: 978-1-61208-158-8

server, which affords a set of user interfaces for doctors to trace the patient's health conditions on a frequent basis. The system can provide data mining and signal processing solutions running on the central server side. However, if the process capacities meet to requirements for these algorithms, the Hub can also be used for such purposes. The central data server can provide various set of interfaces for other integrated 3rd party systems. A typical Telemonitoring System and its dataflow are illustrated in Figure 1. As it can be seen, several kinds of data model are presented in a Telemonitoring System. In general, different data models are used on the Hub and on the server side although they work with the same set of data. Developers also have to deal with the mappings to data models of integrated systems. In this paper, we will focus on the data model of the central server, which provides the basis for the communication and the dataflow between the server and the Hubs, the user interfaces and the integrated systems.

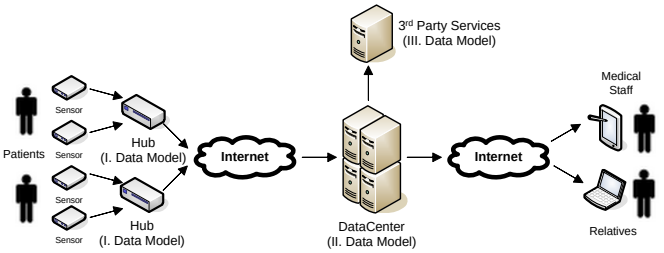


Figure 1. Dataflow in Telemonitoring Systems

Although all Telemonitoring Systems follow this common structure, a variance can be observed between them. This is mainly caused by the fact that these systems have different target diseases to monitor. In addition, for each disease a well-defined and peculiar set of sensors are needed to be integrated to the system. The discrepancy of the devices can lead to disease or sensor specific data models [7]. Accordingly, the integration and expansion of these systems is difficult or even impossible. On the other hand, a generic data model that does not contain such constraints is hard to design and develop because a fully detailed specification is required. This specification is time-consuming and in several cases the assumptions of the projects do not allow such pursuits.

These considerations reveal how difficult it is to design a sophisticated data model and as such, data modeling is one of the most critical proportions of the software design phase, although it signifies only a minor part of the total development effort. The data model also has an impact on system flexibility, reusability, implementability, performance, and on integration with other systems. A not warily designed data model results in various functional deficiencies of the final system. Modifying the requirements or adding previously unidentified requirements to the system specification in the later stages of the software development process costs multiple times more and it can dramatically increase the overall development costs. The completeness and the quality of data models are particularly important in Telemonitoring Systems, where in production usage human health or even human lives are at stake.

III. DATA MODELING PRINCIPLES AND QUALITY METRICS

In practice, it is relevant to differentiate the data model, which is the final product and data modeling, which is the process used to build the final product. In spite of both are substantial, improving the process quality results in a higher and more sustainable data model quality level, as it concentrates on defect prevention rather than detection. Moreover, this also means that a good data modeling process can reduce costs because data quality issues can be resolved at the earlier stages of the development [8]. A few process proposals were laboured out, but they were mainly advised as a toolkit for data modeling experts, rather than as a rigid guideline to be followed [5][9]. Although a set of data modeling patterns are represented in [10] based on real systems, only the semantic and conceptual correctness of these models are defined, the quality of them regarding numerical metrics is ignored. As it says, “patterns are a starting point, not a destination”. However, if the main principles are observed and enough time is allocated for analyzing not only the adaptable patterns and the data model but also the modeling process, a better data model quality can be achieved.

A continuous assessment by quality metrics is required to ensure the best data model quality during software development. The metrics help improve the quality of the data model, choose between alternative models, and improve the modeling process. Several metrics were laboured to measure data model quality [11]. One of the most comprehensive collections is assembled in [12], where 29 metrics were identified and organized around 8 main Quality Factors (Figure 2). Each Quality Factor also refers to a responsible group of stakeholders who are involved in evaluating the metrics related to the given factor.

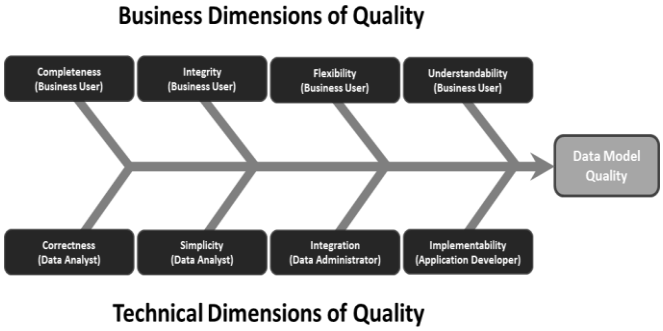


Figure 2. Data Model Quality Factors [12]

The definitions of the quality factors defined in [12], which were the ground for evaluating our models, are:

- Correctness was defined as whether the model conforms to the rules of the data modeling technique (i.e., whether it is a valid data model).
- Completeness refers to whether the data model contains all information required to support the required functionality of the system.
- Integrity is defined as whether the data model defines all business rules, which apply to the data.

 ISBN: 978-1-61208-158-8

- Simplicity means that the data model contains the minimum possible entities and relationships.
- Flexibility is defined as the ease with which the data model can cope with a business and/or regulatory change.
- Integration is defined as the consistency of the data model with the remaining data of the organization.
- Understandability is defined as the ease with which the concepts and structures in the data model can be understood.
- Implementability is defined as the ease with which the data model can be implemented within the time, budget, and technology constraints of the project.

Although these quality factors define a broad set of metrics for evaluating data models, in [13] these quality factors were used extensively in real scenarios and their usefulness was measured, where the empirical validation proved that only three of them were subservient in practice. In addition, two new metrics the reuse level and the number of issues by quality factors were found. Furthermore, sometimes the subjective ratings of the data modeling experts were more useful to comprehend overall data model quality, and the form of textual descriptions about quality issues disclosed a better view of current defects as these were listed the issues themselves rather than a quantitative measurement. In theory, metrics can measure quality, but in practice they are not definitely useful and that is why the opinions of data modeling experts about the importance of each quality metric are divided. The only principle that is accepted by a wide group of experts is the following: the usefulness of a quality metric can be justified if its added value is more than the cost of the effort to measure it.

According to our Telemonitoring Systems and their Living Lab based experiments, we could measure some other, but relevant metrics. These were performance, productivity and user satisfaction.

IV. EXAMINED DATA MODELS

In this section, the two examined data models will be presented. We will focus on a small but critical point of the overall models that can be easily compared as it is responsible for the same role in the same domain: storing and managing medical data in a Telemonitoring System. We illustrate each model using UML class diagram notation, as the structure of them is more relevant in our case than their semantic meanings.

The long-term project is called ProSeniis [14][15] and the mid-term project is called Medistance [16]. In both systems a broad range of sensors were integrated and several kinds of data were stored. In the ProSeniis project the final system had more than 150 user interfaces and ~170000 lines of code provided the overall functionality. 155 entities were responsible for storing all kinds of data. Several developers were involved in the development of the system for a two-year time period and the average of allocated person-months (PMs) per functionality was 1,5. Regarding the Medistance project, ~65000 lines of code and ~50 user interfaces were laboured out in about half a year (0,7 PMs/functionality). In the persistence layer of the system 41 entities were defined to store the data. Both persistence layers were designed and developed on the top of Hibernate [17].

Each project was managed with a project management tool, called Trac [18]. In the Trac system the full development processes were tracked in the form of structured tickets commented on a daily basis. Moreover, this kind of reporting was also helpful to evaluate the differences in productivity that contains design, development, and maintenance as well. Since we have finished the two mentioned projects, we have laboured out a more sophisticated methodology and a plug-in tool to measure productivity during the whole development process [19]. Based on the results, in the future hopefully we will be able to provide a more detailed overview on efficiency of different development aspects.

The overall model and functionality of the systems are out of the scope of this paper. We selected the simplest physiological data: blood pressure, blood-glucose, and bodyweight to represent the key differences between the models in the following sections.

A. Problem Specific Data Model

In our mid-term project, we were focusing only on the e-Health domain and its characteristics as much as the budget and time constraints allowed. The data model for storing and managing the physiological data described above was simply defined by a subtype/supertype hierarchy using inheritance. The parent entity is the *Data*, which contains the common attributes assigned for all types of data. Each derived entity represents a data type and its peculiar attributes. In Figure 3, the UML class diagram for this data structure can be seen.

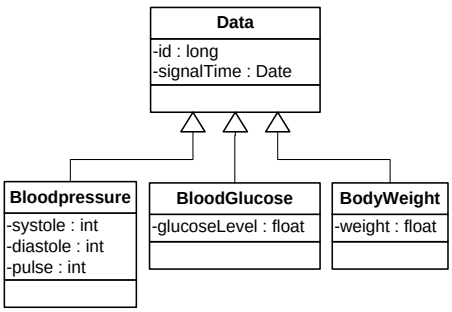


Figure 3. Problem Specific Data Model

The simple and clear data model served as a perfect basis to build up a fully broad system where all the functionalities were exactly fitted to the user requirements, and the doctors received a set of the accustomed views of the collected data, which were familiar to them from other medical systems. Furthermore, the predefined views were feasible to give a comprehensive insight into the patients' current health status along with ease-of-use user interfaces.

B. Generic Data Model

In the long-term project, the goal was to build a generic Telemonitoring System where the type of devices and the measured data the system deals with were not preliminarily

 ISBN: 978-1-61208-158-8

defined and limited to a number of sensors. The ability to simply or even in runtime add sensors to the system had the main impact on the design of the data model. Four entities were identified to store data. The *Datatype* entity defines the structured hierarchy of data types and for each entity a set of *DatatypeDescriptor* entities are appointed the attributes that belongs to the type. The *GenericData* entity can be imagined as an instance of the *Datatype* where the *GenericDataAttribute* instances hold the current values for attributes (Figure 4). The *Datatype* and *DatatypeDescriptor* describe the schema for the data type, e.g., the blood pressure data type has three attributes and all of them are integer values. The *GenericData* instances can be interpreted with the help of its *Datatype,* and the attributes can be processed as the *DatatypeDescriptors* circumscribe them.

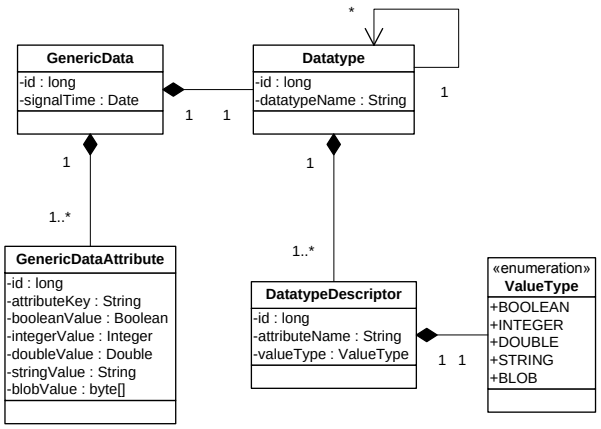


Figure 4. Generic Data Model

The model fulfilled all the predefined requirements. In addition, the generic data management was complemented with the ability to configure the measurements, signal processors and their dependencies in runtime by the end-users. A general data charting and representation could be built upon the data model where the doctors were free to define diagrams combining several measured and derived data with selecting a patient, a data type, and an attribute of the data type. The *Datatype* and *DatatypeDescriptor* entities were enough to define and render the diagrams containing the appropriate *GenericData* instances and the values of their attributes in a generic way. A similar data model is described in [7].

V. EVALUATION

The examination of each data model was based on the Quality Factors described above. The evaluation of *correctness* and *completeness* of the data models are out of the scope of this paper, but the Living Lab based experimentation pointed out that both of the models were actually correct and fulfilled the functional requirements.

Regarding *integrity*, the problem specific data model performs better as in the generic model some of the business rules are missing. Precisely, in the generic model the constraints on the values of the attributes are not enforced, which can cause inconsistency in the database and incorrect data could be stored. Such constraints are not allowed to specify on the model since different attributes of different data types have been stored in the same database column if the types of the attributes have been the same. These rules must be enforced in the higher layers of the system, e.g., in the data access layer or by triggers or interceptors. However, this is not a critical issue as the constraint validation can be supported to be able to automatically evaluate the rules by extending the *DatatypeDescriptor* entity with some attributes according to the constraints for the given data attribute. To stay with the models presented above, if we want to store blood pressure, blood-glucose, and bodyweight data 5 constraints are missing currently. This number is growing in line with the number of the data types defined to store. In the problem specific data model all these constraints can be exactly defined because each attribute is stored in a separate column. Regarding this integrity issue, the key difference can be observed between the architecture of each persistence layer based on these data models. Data validation can be found at a different level as it can be seen in Figure 5 and in Figure 6.

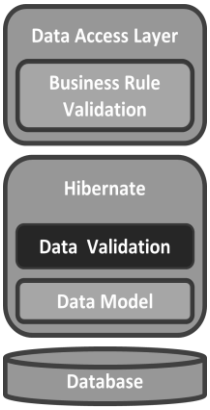


Figure 5. Specific Model based Persistence Layer

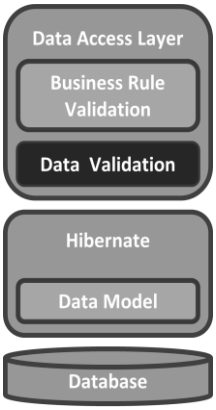


Figure 6. Generic Model based Persistence Layer

The *simplicity* of the data model can be defined by the number of entities and relationships presented in the model. The inheritance does not improve this number as in a physical database it is often mapped to a single table where all instances of the derived classes are stored and the derived entities represent only subcategories within a single construct. The complexity of the problem specific data model is 1, while the complexity of the generic data model is 8 (4 entities and 4 relationships). As it could be expected, the problem specific data model is simpler than the generic one, but this rate is invariant and does not depend on the number of data types defined to be stored.

The number of elements in the model, which are subject to change in the future and the cost estimation to implement the changes are defined as the metrics to evaluating *flexibility*. Although the probability of change is small in the structure of medical data, alteration in the auxiliary attributes could occurred, e.g., the current models do not deal with measure units. Also, it is possible that new data types appear later on to extend the system. The generic data model can handle these changes without any modification on the model, but in the problem specific data model all the 4 entities could be expected to be modified in the future.

 ISBN: 978-1-61208-158-8

The *reusability* of the data models in a Telemonitoring System or even partially in any kind of system in the e-Health domain is unambiguous and perfectly fits into this scope. However, the generic data model is not specialized only for e-Health development purposes, the overall data model is reusable in a different domain, e.g., in the field of agriculture to store data about plant status in a greenhouse. It means 4 entities are reusable from the generic data model in contrast with the problem specific data model where this number is 0 when we leave the scope of e-Health systems.

The *integration* of these models with other systems were out of scope in our R&D projects, but it is apparent that the generic model is more flexible and reusable as the previous metrics pointed out above, so it could be easier to provide interfaces and adapters towards other systems than implementing them over the problem specific data model. Furthermore, if the integration process requires some modifications in the data model, it could be seamlessly done in the generic data model.

For the *understandability* of a data model, not a voluble, quantitative metric exists. It is mostly the subjective opinion and rating of the users and the developers that provide information about this quality factor. Regarding our experience, the effort required to understand the generic data model was twice as much than the effort needed for the problem specific model. The ratio of entities and attributes also shows this deviation, where the metric is $^4/_{14}$ for the generic data model and $^4/_7$ for the problem specific model. The key difference is that the generic model is unable to interpret its context without concrete, domain specific examples.

According to the *implementability* the problem specific model is easier to implement, the only technical risk is the ability to provide an adequate solution to handle inheritance. The generic data model has more entities and relationships and the developers also have to deal with the integrity issues along with the data model implementation. The risk and effort are higher in the case of generic models, but not considerably.

Additionally, the *productivity* can be mentioned as a partial subcategory for implementability where the maintenance of the model is also engaged. In the design and implementation phases the problem specific data model was laboured and developed in $^1/_3$ less time than the generic model (the overall data model). However, the costs of maintenance are much higher in the case of the specific model. If changes affect the data model, all system layers have to be modified. Our experience was that it took $^1/_{10}$ less time to implement modifications in the system that was based on the generic model.

During the Living Lab experimentations both systems were applied in production usage. The clinical trial allows us to monitor the *performance* of the system and the model within real conditions. Although both systems were successfully used in the field of e-Health, the performance of the generic data model based system was continuously but not significantly falling back as the number of stored data increased. In Figure 7 the performance of each model can be seen. The diagram shows the required time in millisecond to provide a patient's blood-pressure data for one month along with the size of the database (number of stored data). The performance was monitored during the regular usage of the charting modules in each system.

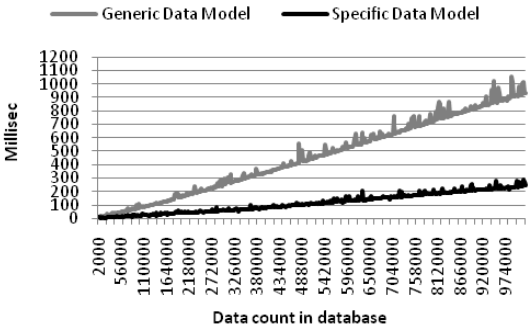


Figure 7. Performance Measurement on Data Models

To summarize the result of the evaluation, an overview of the Quality Factors is presented in the form of a Polar chart in Figure 8. The values are based on the subjective ratings of data modeling experts and the quantitative metrics described above. The chart helps with the conceptual representation of the overall quality and gives a comprehensive view to appraise the models.

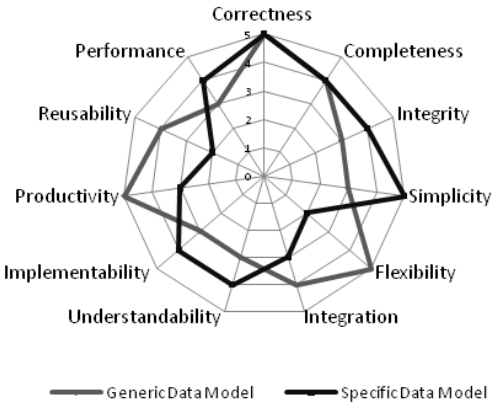


Figure 8. Overview of Data Model Quality Factors

Each model has its advantage in some of the factors however the overall rating is nearly the same. Moreover, there are unambiguous correlations between the Quality Factors [8], e.g., the simpler model is more understandable or the more flexible model is more reusable, which contributes to a fairly distinct set of factors that characterize the benefits of each model.

VI. USER ACCEPTANCE AND SATISFACTION

The benefits of the generic model definitely support the software developers. Although it takes more efforts in the design phase, in the overall development process and after the release the modifications could be easier to adapt. In addition, the flexibility and reusability factors are more relevant for software development. However, in practice the generic software is not the best solution to provide a product, which is

 ISBN: 978-1-61208-158-8

expected to fully cover all user requirements. In our experience, the generic user interfaces where the doctors could build up the charts and views on their own were not as successful as it could be expected. Ironically, it turned out that regarding the predefined views the doctors were more satisfied and they preferred to use that kind of user interface. The generic views require more competence from the users, which they cannot accept easily.

VII. CONCLUSION

In this paper, we overviewed the importance of the data model in a Telemonitoring System and we summarized the results of the evaluation of two different data models designed and developed in the field of e-Health. We revealed and compared the benefits and drawbacks of each model. While the generic data model is more feasible for software developers and can be adopted in long-term development procedures, the problem specific data model allows a rapid development with valuable results in short- or mid-term projects. Currently, it depends on the context and the conditions of the project which data modeling aspect would be better as a balance can be identified between the two sides (Figure 9).

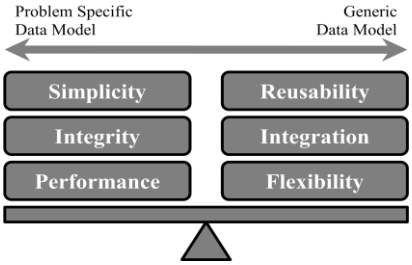


Figure 9. Balance of Data Modeling Aspects

Setting up adequate weightings along the quality factors while analyzing and identifying the key requirements of the system and the available resources could be helpful to decide which data modeling aspect will be the most convenient. However, based on our experience a specialized user interface based on a generic data model could be an acceptable intermediate solution in any case.

ACKNOWLEDGMENT

The work presented was partly funded by the National Innovation Office, Hungary (project No. OM-00191/2008-AALAMSRK 'ProSeniis'), Telenor 19/55 1I066, Nokia Komarom 19/55 1C117 and GOP-1.1.2-07/1-2008-0007.

REFERENCES

[1] M. Suh, L. S. Evangelista, C. Chen, K. Han, J. Kang, M. K. Tu, V. Chen, A. Nahapetian, and M. Sarrafzadeh, "An automated vital sign monitoring system for congestive heart failure patients," Proceedings of the ACM international conference on Health informatics - IHI '10, Arlington, Virginia, USA: 2010, pp. 108.

[2] M. Sarriegui, G. Sáez, H. Pérez, M. Elena, R. Cros, E. Brugués, A. Leiva, G. Aguilera, and J. Enrique, "Mobile Telemedicine for Diabetes Care," Mobile Telemedicine A Computing and Networking Perspective, CRC PRES, Taylor & Francis Group; ISBN: 9781420060461, 2008.

[3] J. G. Cleland, A. A. Louis, A. S. Rigby, U. Janssens, and A. H. Balk, "Noninvasive Home Telemonitoring for Patients With Heart Failure at High Risk of Recurrent Admission and Death: The Trans-European Network-Home-Care Management System (TEN-HMS) study," Journal of the American College of Cardiology, vol. 45, 2005, pp. 1654–1664.

[4] S. Scalvini, M. Vitacca, L. Paletta, A. Giordano, and B. Balbi, "Telemedicine: a new frontier for effective healthcare services," Monaldi Arch Chest Dis, vol. 61, 2004, pp. 226–233.

[5] Graeme C. Simsion and Graham C. Witt, Data modeling essentials, Morgan Kaufmann; ISBN: 978-0126445510, 2004.

[6] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," Computer Networks, 2010.

[7] J. Cai, S. Johnson, and G. Hripcsak, "Generic Data Modeling for Home Telemonitoring of Chronically Ill Patients," AMIA, Inc., 2000.

[8] D. L. Moody, G. G. Shanks, and P. Darke, "Improving the Quality of Entity Relationship Models," Lecture Notes in Computer Science, vol. 1507/1998, Springer-Verlag Berlin Heidelberg, 1998, pp. 255−276.

[9] M. West and J. Fowler, "Developing High Quality Data Models," EPISTLE, 1996.

[10] M. Fowler, "Analysis Patterns: Reusable Object Models," Addison-Wesley Professional, ISBN: 978-0201895421, 1996.

[11] M. Piattini, M. Genero, and C. Calero, "Data model metrics," Handbook of Software Engineering and Knowledge Engineering, vol. 2, 2002, pp. 981–02.

[12] D. L. Moody, "Metrics for Evaluating the Quality of Entity Relationship Models," Lecture Notes in Computer Science, vol. 1507/1998, Springer-Verlag Berlin Heidelberg, 1998, pp. 211−225.

[13] D. L. Moody, "Measuring the quality of data models: an empirical evaluation of the use of quality metrics in practice," Proceedings of the Eleventh European Conference on Information Systems, ECIS, 2003.

[14] http://www.proseniis.com; 26.04.2011, "ProSeniis Home Page."

[15] I. Vassányi, G. Kozmann, B. Végső, I. Kósa, T. Dulai, D. Muhi, and Z. Tarjányi, "Alpha: multi-parameter remote monitoring system for the elderly," MIE'2010, Cape Town, South Africa, 2010.

[16] http://www.medistance.hu; 26.04.2011, "Medistance Home Page."

[17] http://www.hibernate.org; 26.04.2011, "Hibernate."

[18] http://trac.edgewall.org; 26.04.2011, "The Trac Project."

[19] G. Tóth, A. Z. Végh, A. Beszédes, and T. Gyimóthy, "Adding Process Metrics to Enhance Modification Complexity Prediction," International Conference on Program Comprehension, 2011

 ISBN: 978-1-61208-158-8

A Rewriting Logic-based Meta-Model for Design Patterns Formalization

Halima Douibi, Kamel Boukhelfa, Faiza Belala
LIRE Laboratory. University Mentouri of Constantine
Constantine, Algeria
{douibi_halima, boukhelfakamel}@yahoo.fr, belalafaiza@hotmail.com

***Abstract*— Informal description of design patterns is adopted to facilitate their understanding by software developers. However, these descriptions lead to ambiguities limiting their correct usage in support tools. Hence, there is a need for formal specification of the design patterns to ensure their successful application. In this paper, we propose a new formalization of design pattern while using a meta-model, based on rewriting logic. The meta-model is encoded in Maude to provide an executable framework allowing experimentation of design patterns models and their formal analysis. Indeed, the relevant elements that constitute a design pattern solution are formally deduced from this formalization.**

Keywords-Design patterns; Meta-model; Rewriting logic; Maude.

I. INTRODUCTION

A design pattern expresses solution of a known and recurrent problem in a particular context. It is applied in object programming software to improve the quality of the expected system [3].

The design patterns are generally described by using a combination of textual descriptions, object oriented graphical notations such as UML's diagrams and sample code fragments. This informal description is adopted to facilitate their understanding by software developers. However, these descriptions lead to ambiguities limiting their correct usage in support tools. Hence, there is a need for formal specification of the design patterns to ensure their successful application. Indeed, this precise and rigorous description permits to achieve the following goals:

- a full understanding of the patterns semantics
- a formal analysis to resolve some issues such as patterns duplication, refinement, disjunction and composition
- a development of the patterns integration into CASE tools.

The formal approaches to design pattern specifications are not intended to replace existing informal approaches, but to complement them.

In this work, we propose a rewriting logic-based meta-model to formalize design pattern solutions and their instantiations. Our proposed meta-model includes all the common elements of design patterns, so any design pattern can be expressed in terms of this meta-model. It provides a high level of abstraction that will cover all the features of design patterns, it allows a generic representation that is used to produce automatically any design pattern specification. Rewriting logic is identified as a semantic basis of our approach since it constitutes an unified semantic framework for many concurrent models. Besides, it has an important property which is reflection allowing powerful meta-programming uses. Intuitively, a logic is reflective if it allows to express at object (or data) level a meta-level (or type level) description. It is used extensively in our meta-model implementation represented as a rewrite logic theory [5]. Hence, we show how this meta-model can be implemented in Maude language. This implementation exploits fully the flexible parser of Maude language and its facilities to define the concrete syntax of the relevant features of our meta-model and its possible analysis.

The rest of this paper is organized as follows: in Section 2, a synthesis on related work for design patterns formalization is given. Then, we present in Section 3 the key concepts of the rewriting logic and its practical Maude language. Section 4 describes on one hand, the proposed meta-model of design patterns, and on the other hand, its encoding in Maude language. Furthermore, an illustrative example is given to elucidate the main idea of our approach. Section 5 concludes this work and presents its perspectives.

II. RELATED WORK

Several attempts to formalize design patterns have been proposed. In this section, we present a brief survey about them focusing especially on the specification formalisms dealing with structural and behaviour aspects of design patterns.

In [9], BPSL (Balanced Pattern Specification Language) language is proposed. This language uses a subset of first-order logic (FOL) to formalize structural aspect of patterns, while the behavioural aspect is formalized in TLA (Temporal Logic of Actions). The first-order logic is justified by its simplicity to express relations between pattern participants as predicates.

In [4], the author presents a use of formal language LePUS (LanguagE for Patterns Uniform Specification) to describe design patterns. LePUS is a fragment of the monadic high-level order logic using a limited vocabulary of entities and relations. A LePUS instruction is formed by a list of participants (classes, functions or hierarchies) and a list of relations between these participants. A program is represented by a model M which is a pair <P, R> where P is the universe of the basic entities (classes and functions) and R=R1,…, Rn is the set of the relations between these entities. These relations are deduced by generalization of all the

 ISBN: 978-1-61208-158-8

existing basic relations between participating entities in the GOF patterns (GOF for "Gang Of Four") [3]. Hence, a design pattern is described by HOL formulae which are accompanied by a graphic representation in order to facilitate its understanding.

Other research works deal with the issues of design patterns integration in CASE tools. We can cite [2] about DPML (Design Pattern Modeling Language) which defines a meta-model and a notation for specifying design pattern solutions and solution instances within object models.
The meta-model defines a logical structure of objects DPML which can be used to create models of design pattern solutions and design pattern solution instances, while the notation describes the diagrammatic notations used to represent visually the models.

At present, DPML allows only specifying the structural aspect of pattern design and no indications are mentioned about the composition and the verification of design patterns. However, instantiation is achieved by mapping from the pattern specification to its realization in a UML design model. The most interesting element of DPML is that it uses a simple set of visual abstractions and readily lends itself to tool support.

Unlike our approach, the most emerging ones are founded on hybrid models and tackle formalization of only some concepts of design patterns which are closed to the object level.

In the present work, we aim to formalize design patterns using a new meta-modeling approach, in which the meta-model represents a part of the global standardized UML meta-model as described by [1]. This meta-model is then integrated in rewriting logic framework.

Our approach differs mainly from the above cited works by the use of a common formalism to specify both the structural and behaviour aspects of design patterns. Moreover, the use of the meta-model concept in design patterns formalization permits to describe all the design pattern features at a same high level of abstraction. In addition, with the encoding of our meta-model in Maude, we obtain executable programs that can be subject of several analysis and verifications.

III. REWRITING LOGIC AND MAUDE

Rewriting logic is known as being logic of concurrent change taking into account the state and the calculus of the concurrent systems. It was shown as a unifying semantic framework of several concurrent systems and models [5][6]. In this context, we can cite without being exhaustive, the labelled transitions systems, Petri nets, CCS, etc.

In rewriting logic, a dynamic system is represented by a rewriting theory $\Re=(\Sigma, E, R, L)$ describing the complex structure of its states and the various possible transitions between them. In rewriting theory definition, (Σ, E) represents an equational membership theory, L is a set of labels and R is a set of labelled conditional rewriting rules. These rewriting rules can be of the following form:

$$(\forall X)\, r{:}\, t \to t' \text{ if } \bigwedge_{i\in I} p_i = q_i \wedge \bigwedge_{j\in J} w_j : s_j \wedge \bigwedge_{l\in L} t_l \to t'_l$$

where r is a labeled rule, all the terms (p_i ,q_i ,w_j ,s_j ,t_l ,t_l') are Σ-terms and the conditions can be rewriting rules, membership equations in (Σ, E), or any combination of both. Given a rewriting theory, we say that $\mathcal{R}$ implies a formula [t]→[t'] if and only if, it is obtained by a finite application of the following deduction rules :

1. Reflexivity:
For each term [t] ∈ $T_{\Sigma,E}(X)$, $[t] \to [t]$
where $T_{\Sigma,E}(X)$, is the set of Σ-terms with variables.

2. Congruence:
For each operator $f \in \sum_n$, n ∈ N,

$$\frac{[t_1] \to [t_1'] \cdots [t_n] \to [t'_n]}{[f(t_1,\ldots,t_n)] \to [f(t'_1,\ldots,t'_n)]}$$

3. Replacement:
For each rewriting rule,
r :[t(x1,…,xn)] → [t'(x1,…,xn)] in R :

$$\frac{[w_1] \to [w'_1] \ldots [w_n] \to [w'_n]}{[t(\overline{w}/\overline{x})] \to [t'(\overline{w'}/\overline{x})]}$$

4. Transitivity:

$$\frac{[t_1] \to [t_2]\,[t_2] \to [t_3]}{[t_1] \to [t_3]}$$

Rewriting logic is also a reflexive logic, i.e., aspects of its meta-theory can be represented in a consistent way, namely there is a universal theory U in which any finitely presented rewrite theory R (including U itself) can be presented as a term $\overline{R}$, any terms t; t' in R as terms $\overline{t}$, $\overline{t'}$ and any pair (R, t) as a term $\langle \overline{R}, \overline{t} \rangle$, so that the following equivalence is established : $R \vdash t \to t' \Leftrightarrow U \vdash \langle \overline{R}, \overline{t} \rangle \to \langle \overline{R}, \overline{t'} \rangle$

The rewriting logic theoretical concepts are implemented through the Maude language [5][6]. Maude objective is to extend the use of the declarative programming and the formal methods to specify and verify critical and concurrent systems. Maude program is simple and easy to understand. It represents a rewriting theory, i.e., a signature and a set of rewriting rules. The computation in this language corresponds to the deduction in rewriting logic. Maude integrates also equational and object oriented programming, which are used in our formalisation to describe our proposed meta-model, in a convenient way. Its logical basis facilitates a clear definition of the object oriented semantics and makes it good choice for the formal specification of object oriented systems. In this case, a concurrent system is modeled by a multi-set of objects and juxtaposed messages. Concurrent interactions between objects are governed by rewriting rules.

 ISBN: 978-1-61208-158-8

An object is represented by the term < O : C | a1 : v1 , … , an : vn >, where O is the object instance name of class C, ai, I ∈ 1..n, the object attributes names, and vi, their respective values.

The class declaration follows this syntax:
class C | a1 : s1 , … , an : sn .
Where C is the name of the class and si is the sort of the attribute ai. It is also possible to declare sub-classes to benefit from the class inheritance.

Messages are declared using the keyword "msg". The general form of a rewriting rule in the Maude's object-oriented syntax is:

crl [r] : M1 … Mn < O1 : F1 | at1> … < Om : Fm | atm > => < Oi1 : F'i1 | at'i1 > … <Oik : F'ik | at'ik > M1' … Mp' if Cond .

r is the rule label, Ms, s ∈ 1..n, and M'u, u ∈1..p, are messages, Oi, i ∈ 1..m, and Oil, l ∈ 1..k, are objects, Cond is the rule condition. If the rule is not conditional, we replace the keyword crl by rl and we remove the clause if Cond.

Another important aspect which favors the use of Maude language is its implementation through a running environment, allowing prototyping and formal analysis of concurrent and complex systems.

IV. THE META-MODEL FORMALIZATION APPROACH

In this section, we present our design patterns formalization approach based on the rewriting logic formalism. This executable logic is intended to specify both the structural and behaviour aspects of design patterns contrary to the other formalization approaches which use at least two distinct formalisms. Besides, its reflective feature allows us to reason on design patterns meta models instead of their formal specifications only. Thus, the use of the meta-model concept in design patterns formalization permits to describe all the design pattern features at a same high level of abstraction. In the following, we first define our meta-model for specifying design pattern solutions. Then, we show how to encode it in Maude to obtain executable design pattern models that can be subject of several analysis and formal verifications.

A. The proposed Meta-Model

To formalize design pattern models and their solutions, we suggest to use a generic notation based on the following meta-model (see Figure 1) as it was done by authors of [1]. Our meta-model defines a logical structure of elements involved in the models conception of the design pattern solutions.

We consider a design pattern solution as a collection of elements and constraints on these elements. An element represents a structural significant part of a design pattern solution. In the object oriented context, this can express a class, an operation or an attribute. Constraints represent conditions that must be verified by an element (i.e., class, operation or attribute). Also, they may represent OCL constraints in object oriented framework.

Thus, the core concept of our meta-model, design pattern model (DP Meta-Model), serves to generate a design pattern solution. The other elements are joined with a set of specific relationship.

We have chosen class diagram notation to represent graphically our meta-model (see Figure 1). Classes and UML relationship (agregation, inheritance, composition, etc.) are used to represent respectively elements and relationship of the meta-model. Thus, the transcription of a class diagram in Maude is easily made thanks to the strong correspondence between UML class concept and the one in Maude [8]. We divided the classes of the solution part of design patterns in several parts (elements, operations, attributes, and constraints). Element class in our meta model represent classes in the solution part, operations of a class are represented by the class operation, the attributes of a class are also represented by a class called attribute. We can see in

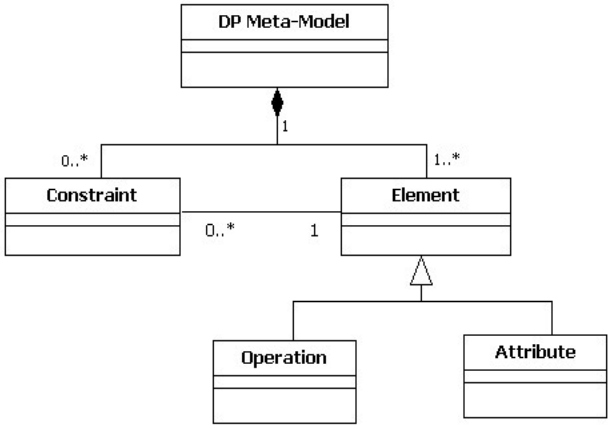


Figure 1. A Meta-model of design patterns

(Figure 1) that the two classes operation and attributes are subclasses of the element class, so each pattern may be composed of a set of elements, operations, attributes and possibly a set of constraints on these elements.

Example: Let us consider as a simple example the Singleton Pattern from [3] (see Figure 2). This pattern is used to ensure that a class has only one instance, and provides a global point of access to it. The solution part of Singleton consists of a single class which contains one attribute called instance and one operation called getinstance().

This pattern can be expressed using our meta-model as it is shown in (Figure 3). In this meta-model there are one main element called singleton which corresponds to singleton class in the solution part of the singleton design pattern, we have also getinstance and instance elements which correspond respectively to the getinstance() operation and instance attribute in the solution part of the singleton design pattern. Finally, we find Return-unique-instance element which represent the constraint imposed by the design pattern.

 ISBN: 978-1-61208-158-8

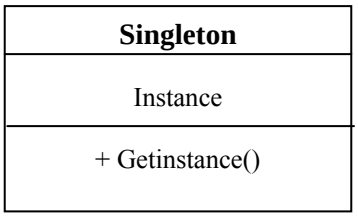


Figure 2. Singleton design pattern

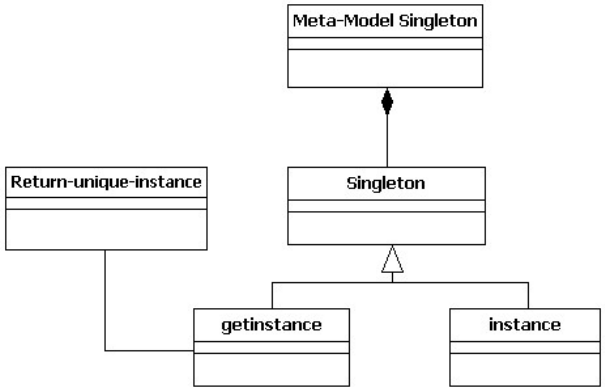


Figure 3. Applying DP Meta-Model to Singleton

B. *Encoding the DP Meta-Model in Maude*

Our proposed meta-model is formalized using rewriting logic model, we adopt object oriented concepts of this logic to define all the components of the meta-model "DP Meta-Model". The proposed model is described in this case, as being a multi-set of juxtaposed objects and messages, where the concurrent interactions between the objects are governed by rewrite rules. Objects represent elements (participants) of the proposed meta-model that may have some behaviours. Then, the semantics of these object interactions is materialized by the defined messages.

We exploit rewriting logic as a unique semantic formalism for specifying and checking design patterns and their solutions. Thanks to this formalization we lean on the category model to give precise and sufficient semantics to behaviour aspects in design patterns. Besides, this high level specification constitutes an executable one, it allows formal analysis using a particular well-founded language Maude having a proof and prototyping environment.

So, we first give the Meta-model class in our global object oriented Maude module Meta-Model-DP with a specific attribute called Name-ID of sort QID.

```
omod Meta-Model-DP is
  Including QID
  Class Meta-model | NameID : QID .
  ...
endom
```

This will define configurations of concurrent models of possibly several design patterns. Rewriting theory describes static and dynamic aspects of these models. Elements and constraints components in the Meta-model are respectively declared as classes called D-element and D-constraint, having one attribute Meta-Model to relate all elements and also constraints of a given Meta-model. The second attribute of D-constraint class is defined to indicate which element is concerned with this constraint:

```
Class D-element | Meta-Model : oid .
Class D-constraint | Meta-Model : oid,
Element : oid .
```

Elements of type Operation or Attribute in the design patterns meta-model are defined as sub classes of D-element class with an attribute called Element to indicate the operation or the attribute to which element they are attached:

```
Subclass D-operation | Element : oid <  D-
element .
Subclass  D-attribute | Element : oid < D-
element .
```

Thus, we have defined the essential parts of elements or constraints structures involved in the meta-model of design patterns. In addition, we are able to provide an object-message fair rewriting strategy that is well suited for executing objects of Meta-Model DP configurations. For lack of paper space, we have limited our work to the following set of messages which have as role to deal only with the different constructs of the meta-model. For instance, the first message is called get-element, it has one argument of type oid (of class D-element):

```
msg _get-elements :    Oid → Msg .
msg _get-constraints : Oid → Msg .
msg _get-operations_:  Oid Oid → Msg .
msg _get-attributes_:  Oid Oid → Msg .

rl [get-elements]:
< O1 : Meta-model | NameID : S1 >
< O2 : D-element | Meta-Model : O1 > O1get-
elements   =>
< O2 : D-element | Meta-Model : O1 > .

rl [get-constraints]:
< O1 : Meta-model | NameID : S1 >      <
O2 : D-element | Meta-Model: O1 >    < O3 :
D-contraint | Meta-Model : O1 , Element: O2>
O1get-contraints   =>
 < O3 : D-contraint | Meta-Model : O1 ,
Element: O2> .

rl [get-operations]:
< O1 : Meta-model | NameID : S1 > < O2 : D-
element | Meta-Model: O1 >  < O3 : D-
operation | Meta-Model : O1 , Element: O2>
 O1get-operations O2
=> < O3 : D-operation | Meta-Model : O1 ,
Element: O2> .
rl [get- attributes]:
```

```
< O1 : Meta-model | NameID : S1 > < O2 : D-
element | Meta-Model: O1 >  < O3 : D-
attribute | Meta-Model : O1 , Element: O2>
 O1get-attributes O2
=> < O3 : D-attribute | Meta-Model : O1 ,
Element: O2> .
```

These rules are examples of synchronous message passing rules involving one or more objects and one message on the left-hand side. In these examples new objects are created but no new messages are sent. Another rule type involving the joint participation of at least two meta-models of design patterns may be naturally defined in the same way. Note that the multiset structure of the configuration provides the top-level distributed structure of the design patterns meta-model and allows concurrent application of these rules

The proposed model is generic enough, it may be easily enriched to take into account specific patterns as for example those of [3]. We add to the Meta-Model-DP module some constants (operators and equations) to instantiate our meta-model to a given design patterns. We take the same example of Singleton pattern to show what we need to declare in this new module called for this case DP-SINGLETON (see Figure 4). The constant singleton designate a given configuration specifying the design pattern. This configuration is defined through an equation which contains a set of objects that form the pattern participants: Sing, S-class, instance, get-state and S-constraint .

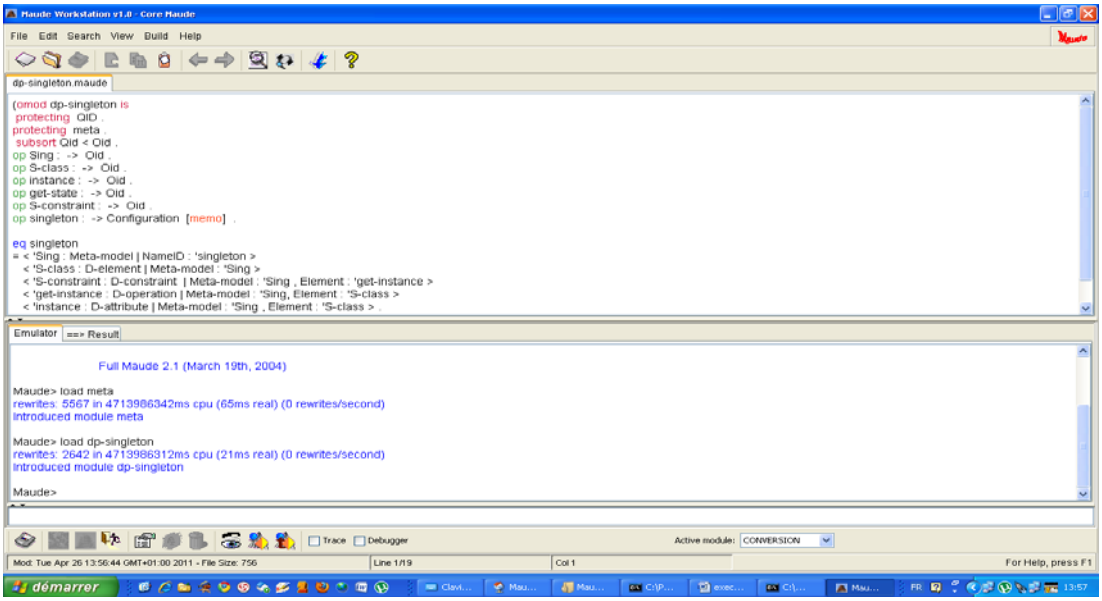


Figure 4. Meta-model module and DP-singleton module

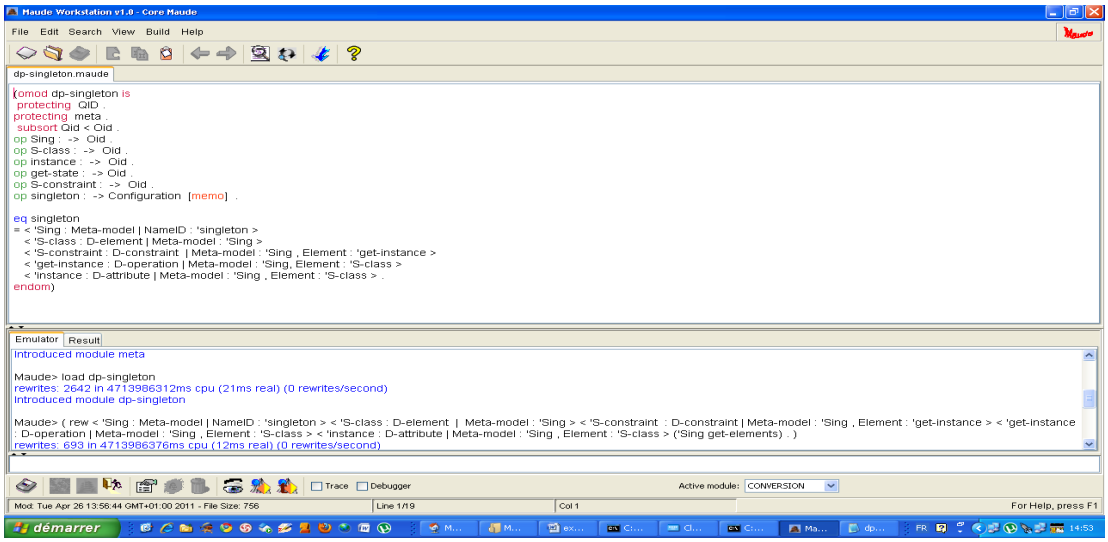


Figure 5. Rewriting configuration example

 ISBN: 978-1-61208-158-8

C. Formal Analysis

The successful implementation of the different modules, previously proposed has been done in Maude workstation 1.0 environment implemented in JAVA. The syntactic verification is implicitly done when the module is loaded (see Figure 4).

We can for instance rewrite any configuration consisting of participants (elements and constraints) of a given design pattern model (or several ones) and some messages with the rewrite command of Maude engine. (Figure 5) presents an example of rewriting a simple configuration to get the elements of the Sing design pattern (Singleton).

This model will enable developers to detect erroneous behaviors and contradictions in the meta models of design patterns. Its main advantage is the clear distinction between the two concerns static and dynamic ones. Thus, firstly we describe the static aspects of the meta-model using Maude objects through their classes. Then, we proceed to the enrichment of this module to ensure its behaviour description. The execution semantic of a model composed of possibly some communicating design patterns models is naturally defined thanks to rewrite rules concurrent execution and Maude module operations. Besides, we can exploit the META-LEVEL predefined Maude module to get the meta representation of these both declared modules and terms (objects and configurations). Thus, we may check if a given design pattern solution respects its pattern or no thanks to metaReduce and metaApply commands.

V. CONCLUSION

The use of formal methods to design patterns is an effective means to improve the reliability and the quality of complex systems. The objective of this work was to adapt one of these methods, largely mastered due to its widespread use in our recent research works, to design patterns model specification and analysis, so that the system development depending of these design patterns solutions can benefit from it.

We have proposed a new approach to formalize design patterns. First we have suggested a meta-model for all design patterns [3]. We considered a design pattern solution as a collection of a participants and constraints on these participants. Our meta-model was defined as a oriented object Maude module, in which all the components of the meta-model are defined as classes. Design patterns can be instantiated from this module. Obtaining the different patterns as dynamic configurations. This will allow us to have multiple patterns in a common configuration.

In the related work section, we have discussed the originality of our proposed approach relatively to other ones that provide languages to formalize design patterns, we have shown through the paper how Rewriting logic (via Maude) offers an accurate way of specifying a meta-model for a generic design pattern and possibly its solutions and provides good tool support for reasoning about them. Our proposed Object Rewrite Theory based model takes benefits from the underlined formalism to consider both static and dynamic aspects of Design Patterns, so it inherits all theoretical advantages of rewriting logic formalism. We do not have need to prove that rewriting logic is better than other formalisms used in this context such monadic high-level order logic, first-order logic (FOL), temporal logic of actions, etc. Besides, rewriting logic has been shown as a logical framework in which several logic have been already integrated.

In future, we will be interested by the study of behavior associated to more complex operations performed on these patterns models such as the composition, the parameterization, etc. On the other hand, defining and encoding in Maude this design pattern meta-model will facilitate the integration of the design patterns into CASE tools. We will take advantage from our meta-modelling approach and the meta model of Maude to deal with Model-to-model transformations (Moment-MT tool, [7]) as technologies looking for reducing the gaps between platform-independent models (PIMs) and platform-specific models (PSMs).

REFERENCES

[1] A. Boronat. “MOMENT: a formal framework for MOdel manageMENT”. PhD in Computer Science, Universitat Polit`enica de Val`encia (UPV), Spain (2007),

[2] D. Mapelsden, J. Hosking, and J. Grundy, “Design Pattern Modelling and Instantiation using DPML”. The 40th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Pacific 2002), Sydney, Australia.

[3] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. “Design patterns: elements of reusable objectoriented systems”. Addison-Wesley, 1995.

[4] E. Gasparis. “LePUS: A Formal Language for Modeling Design Patterns”, Chapter XVI in Design Pattern Formalization Techniques, IGI Publishing (an imprint of IGI Global) (2007), pp. 357-372.

[5] J. Meseguer. “A Logical Theory of Concurrent Objects and its Realization in the Maude Language”. In G. Agha, P. Wegner, and A. Yonezawa, Editors, Research Directions in Object-Based Concurrency. MIT Press, 1992.

[6] J. Meseguer, “Software Specification and Verification in Rewriting Logic”. In M. Broy and M. Pizka, editors, Models, algebras and logic of engineering software, pp. 133-193. IOS Press, 2003. http://maude.cs.uiuc.edu. 7.12.2011

[7] http://moment.dsic.upv.es/. 7.12.2011

[8] K. Boukhelfa, F. Belala, A. Choutri, and H. Douibi, “For more understandable UML diagrams”. In IEEE/ACS International Conference on Computer Systems and Applications (AICCSA) 2010, pp. 1 – 7.

[9] T. Taibi and D. Ngo. “Formal specification of design patterns-A balanced approach”. In Journal of Object Technology, 2, 4. (2003), pp. 127-140.

 ISBN: 978-1-61208-158-8

Pattern-based Software Design and Adaptation

Hassan Gomaa
Department of Computer Science
George Mason University
Fairfax, Virginia 22030, USA
hgomaa@gmu.edu

***Abstract* - This paper describes how software architectural design patterns can be used to build software systems and how software adaptation patterns can be used to dynamically adapt them at run-time. The software architectural design patterns consist of architectural structure patterns and architectural communication patterns. This paper also describes the concept of software adaptation patterns and how they can be used in software system adaptation**.

Keywords - Software design; Unified Modeling Language (UML); software architectural design patterns; software adaptation.

I. INTRODUCTION

Software design methods have advanced considerably over the past three decades from structured methods for centralized systems to advanced design methods for distributed applications and product lines. Recent developments are the emergence of component-based design, software product line design, service-oriented architectures, and applying software architectural and design patterns in the design process.

This paper describes how software architectural patterns can be used to help build and evolve software applications. The approach involves integrating software architectural patterns into a model-driven software development process. The patterns consist of architectural structure patterns and architectural communication patterns. This paper then describes how software architectural patterns can be used for software adaptation.

The paper is organized as follows. Section II describes software architectural design patterns. Section III describes how these software architectural design patterns are applied and integrated to create and evolve software architectures for software applications. Section IV describes the role of software adaptation in software development. Section V describes the concept of software adaptation patterns while Section VI describes several software adaptation patterns that have been developed. Section VII describes the process of run-time adaptive change management for software adaptation patterns. Section VIII provides concluding remarks.

II. ARCHITECTURAL DESIGN PATTERNS

Software architectural patterns [2, 10] provide the skeleton or template for the overall software architecture or high-level design of an application. These include widely used architectures [1] such as client/server and layered architectures. Design patterns [3] address smaller reusable designs than architectural patterns, such as the structure of subsystems within a system. The description is in terms of communicating objects and classes customized to solve a general design problem in a particular context.

Basing a candidate software architecture on one or more software architectural patterns helps in designing the original architecture as well as evolving the architecture. This is because the adaptation and evolutionary properties of architectural patterns can also be studied and this assists with an architecture-centric evolution approach [7].

There are two main categories of software architectural patterns [10]. Architectural structure patterns address the static structure of the software architecture. Architectural communication patterns address the message communication among distributed components of the software architecture.

Most software systems can be based on well understood software architectures. For example, the client/server software architecture is prevalent in many software applications. There is the basic client/service pattern, with one service and many clients. An example of this pattern is given in Figure 1 in which an ATM Client communicates synchronously with a Banking Service by sending a message and waiting for a response. However, there are also many variations on this theme, such as multiple client / multiple service architectures and brokered client/service architectures. Furthermore, with a client/service pattern, evolution can be introduced by replacing a service with a newer version or adding new services, which are discovered and invoked by clients. New clients can be added that discover services provided by one or more servers.

 ISBN: 978-1-61208-158-8

Many real-time systems [4] provide overall control of the environment through centralized control, decentralized control, or hierarchical control. Each of these control approaches can be modeled using a software architectural pattern. In a centralized control pattern, there is one control component, which executes a state machine. It receives sensor input from input components and controls the external environment via output components. In a centralized control pattern, evolution takes the form of adding or modifying input and/or output components that interact with the control component, which executes a state machine.

In a distributed control pattern, control is distributed among several control components, each of which controls local input and output components (Figure 2). The control components communicate with each other to inform each other of new events, as shown in Figure 2.

Another architectural pattern that is worth considering because of its desirable properties is the layered architecture. A layered architectural pattern allows for ease of extension and contraction [13] because components can be added to or removed from higher layers, which use the services provided by components at lower layers of the architecture.

In addition to the above architectural structure patterns, certain architectural communication patterns [10] also encourage adaptation and evolution. In software architectures, it is often desirable to decouple components. The Broker, Discovery, and Subscription/Notification patterns encourage such decoupling. With the broker patterns, services register with brokers, and clients can then discover new services. Thus a software system can evolve with the addition of new clients and services. A new version of a service can replace an older version and register itself with the broker. Clients communicating via the broker would automatically be connected to the new version of the service.

The Subscription/Notification pattern also decouples the original sender of the message from the recipients of the message, as shown in Figure 3. A client (Operator Interaction) subscribes to receive event notifications from a service (Alarm Handling Service). The Alarm Handling Service will then multicast event notifications, whenever they are received from the Event Monitor, to all subscribing Operator Interaction clients.

III. APPLYING SOFTWARE ARCHITECTURAL PATTERNS

A very important decision is to determine which architectural patterns—in particular, which architectural structure and communication patterns—can be applied in the design of a given software application. It is necessary to decide first which architectural structure patterns can be used for the application and then which architectural communication patterns.

In many applications, architectural patterns, including client/service and control patterns, can be integrated with the layered pattern. Integrating the client/service pattern with the layered pattern involves placing clients at higher layers than services, since clients depend on services. With a centralized control pattern, the control component is placed at a higher layer than the components it controls. With the distributed control pattern, the control components are all at the same level since communication between them is peer-to-peer. With a hierarchical control pattern, since the high level controller sends overall control commands to the lower level control components, it is placed at a higher layer in the hierarchy. Typically, communication patterns are used to facilitate the integration of the architectural structure patterns.

Consider an example of applying and integrating software architectural patterns. A distributed emergency control and monitoring system is to be designed. The layered pattern is applied to facilitate the pattern integration. This system has client/service characteristics in that user interaction clients can request emergency monitoring status and alarm status from emergency monitoring services. This system also has distributed control characteristics, since there are control components at different locations that receive local sensor data, such as fire or smoke sensors, and can output commands to local actuators, e.g., to switch on sirens or sprinkler systems. The system could be constructed by integrating the client/service pattern (see Figure 1) and the distributed control pattern (see Figure 2). Integrating these architectural structure patterns could be achieved by using architectural communication patterns such as a Broker pattern or a Subscription/Notification pattern (see Figure 3).

For this application, the Distributed Controller, Event Monitor, and Operator Interaction components are all clients of the Event Handling Service (see Figure 1) and are therefore placed at higher layers in a layered architecture. The Distributed Controller (of which there are multiple interconnected instances as shown in Figure 2) and Event Monitor components send asynchronous event messages to the Event Handling Service. Operator Interaction uses the subscription/notification pattern (Figure 3) to subscribe to the Event Handling Service, which sends multicast notifications to all subscribing clients for each new event it receives. The integration of these software architectural patterns is depicted in Figure 4.

 ISBN: 978-1-61208-158-8

IV. SOFTWARE ADAPTION

Software adaptation addresses software systems that need to change their behavior during execution. In self-managed and self-healing systems, systems need to monitor the environment and adapt their behavior in response to changes in the environment [12].

Software adaptation can take many forms. It is possible to have a self-managed system which adapts the algorithm it executes based on changes it detects in the external environment. If these algorithms are pre-defined, then the system is adaptive but the software structure and architecture is fixed. The situation is more complex if the adaptation necessitates changes to a software component or more widely to the architecture. In order to differentiate between these different types of adaptation, adaptations can be classified as follows within the context of distributed component-based software architectures:

a) Behavioral adaptation. The system dynamically changes its behavior within its existing structure. There is no change to the system structure or architecture.

b) Component adaptation. Dynamic adaptation involves changing one component with another that has the same interface. The old component has to be dynamically replaced by a new component while the system is executing.

c) Architectural adaptation. The software architecture, consisting of multiple components, has to be modified as a result of the dynamic adaptation. Old components must be dynamically replaced by new components while the system is executing.

Model based adaptation can be used in each of the above forms of dynamic adaptation, although the adaptation challenge is likely to grow progressively from behavioral adaptation through architectural adaptation.

V. CONCEPTS OF SOFTWARE ADAPTATION PATTERNS

The software architecture is composed of distributed software architectural patterns, such as client/server, master/slave, and distributed control patterns (Section III), which describe the software components that constitute the pattern and their interconnections. For each of these architectural patterns, there is a corresponding software adaptation pattern [9], which models how the software components and interconnections can be changed under predefined circumstances, such as replacing one client with another in a client/server pattern, inserting a control component between two other control components in a distributed control pattern, etc.

A software adaptation pattern defines how a set of components that make up an architecture or design pattern dynamically cooperate to change the software configuration to a new configuration given a set of reconfiguration commands. A software adaptation pattern requires state- and scenario-based reconfiguration behavior models to provide for a systematic design approach. The adaptation patterns are described in UML with adaptation integration models (using communication or sequence diagrams) and adaptation state machine models [8, 9].

An adaptation state machine defines the sequence of states a component goes through during dynamic adaptation from a normal operational state to a Passive state (in which it does not initiate any new transactions but completes existing transactions), to a quiescent (idle) state, as shown in Figure 5. Once quiescent, the component is idle and can be removed from the configuration, so that it can be replaced with a different version of the component.

VI. SOFTWARE ADAPTATION PATTERNS

Several adaptation patterns have been developed and are described below:

a) The Master-Slave Adaptation Pattern is based on the Master-Slave pattern [2]. A Master component, which sends commands to slaves and then combines responses, can be removed or replaced from the configuration after the responses from all slave components have been received. Slave components can be removed or replaced after the Master is quiescent.

b) The Centralized Control Adaptation Pattern is based on the Centralized Control pattern, and can be used in real-time control applications [10]. The removal or replacement of any component in the configuration requires the Central Controller to be quiescent.

c) The Client / Service Adaptation Pattern is based on the Client / Service pattern [10]. A client can be added to or removed from the configuration after completing the service request it initiated. A Service can be removed or replaced after completing the current service request.

d) The Decentralized Control Adaptation Pattern is based on the Decentralized Control pattern and can be used in distributed control applications [10]. A control component in this Adaptation Pattern notifies its neighboring control components if it plans to become quiescent. The neighboring components cease to communicate with this component but can continue with other processing.

VII. ADAPTIVE CHANGE MANAGEMENT

Adaptive change management is provided by a change management model [9, 11], which defines the precise steps involved in dynamic reconfiguration to transition from the current software run-time configuration to the new run-time configuration. For each software adaptation

 ISBN: 978-1-61208-158-8

pattern, the change management model describes a process for controlling and sequencing the steps in which the configuration of components in the pattern is changed from the old configuration to the new configuration [5].

A component that needs to be replaced has to stop being active and become quiescent, the components that it communicates with need to stop communicating with it; the component then needs to be unlinked, removed and replaced by the new component, after which the configuration needs to be re-linked and restarted. A dynamic software reconfiguration framework is designed and implemented to initiate and control the steps of the change management model for automatic reconfiguration of the architecture from one run-time configuration to another [5, 8].

For example, if it is necessary to replace one of the control components in Figure 2, the control component would need to transition to a quiescent state in which it has completed its current operation and has notified its neighboring components that it is no longer communicating with them. It can then be removed from the configuration and be replaced with a new component (e.g., with enhanced functionality), which is then activated and resumes execution and interaction with its neighbors.

VIII. CONCLUSIONS

This paper has described how software architectural patterns can be applied and integrated in the design of software applications. The paper has also described how the application can be dynamically adapted at run-time to replace one component with another. It is also possible to create and integrate executable design patterns [14]. Current research is investigating software design and adaptation patterns for software product lines [6], as well as how these patterns can be used to assist with software evolution.

REFERENCES

[1] L. Bass, P. Clements, R. Kazman, “Software Architecture in Practice”, Addison Wesley, Reading MA, Second edition, 2003.

[2] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, “Pattern Oriented Software Architecture: A System of Patterns”, John Wiley & Sons, 1996.

[3] E. Gamma, R. Helm, R. Johnson and J. Vlissides, “Design Patterns: Elements of Reusable Object-Oriented Software”, Addison Wesley, 1995.

[4] H. Gomaa, "Designing Concurrent, Distributed, and Real-Time Applications with UML", Addison-Wesley Object Technology Series, 2000.

[5] H. Gomaa and M. Hussein, “Software Reconfiguration Patterns for Dynamic Evolution of Software Architectures”, Proc. Fourth Working IEEE Conf. on Software Architecture, Oslo, Norway, June, 2004.

[6] H. Gomaa, “Designing Software Product Lines with UML: From Use Cases to Pattern-based Software Architectures”, Addison-Wesley, 2005.

[7] H. Gomaa, “A Software Modeling Odyssey: Designing Evolutionary Architecture-centric Real-Time Systems and Product Lines”, Proc. ACM/IEEE 9th Intl. Conf. on Model-Driven Eng., Lang. and Systems, Springer Verlag LNCS 4199, Pages 1-15, Genova, Italy, Oct. 2006.

[8] H. Gomaa and M. Hussein, Model-Based Software Design and Adaptation, Proc. ACM/IEEE ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems, Minneapolis, MN, May 2007.

[9] H. Gomaa, K. Hashimoto, M. Kim, S. Malek, and D. Menascé, Software Adaptation Patterns for Service-Oriented Architectures, Proc ACM Software Applications Conf. (SAC), Sierre, Switzerland, March 2010.

[10] H. Gomaa, “Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures”, Cambridge University Press, February 2011.

[11] J. Kramer and J. Magee, The Evolving Philosophers Problem: Dynamic Change Management, IEEE Trans. on Software Eng., Vol. 16, No. 11, Nov. 1990.

[12] J. Kramer and J. Magee, “Self-Managed Systems: an Architectural Challenge”, Proc Intl. Conf. on Software Engineering, Minneapolis, MN, May 2007.

[13] D. Parnas, "Designing Software for Ease of Extension and Contraction", in Software Fundamentals, edited by D. Hoffman & D. Weiss, Addison Wesley, 2001.

[14] R. Pettit and H. Gomaa, “Modeling Behavioral Design Patterns of Concurrent Objects”, Proc. Intl. Conf. on Software Eng., Shanghai, China, May 2006.

 ISBN: 978-1-61208-158-8

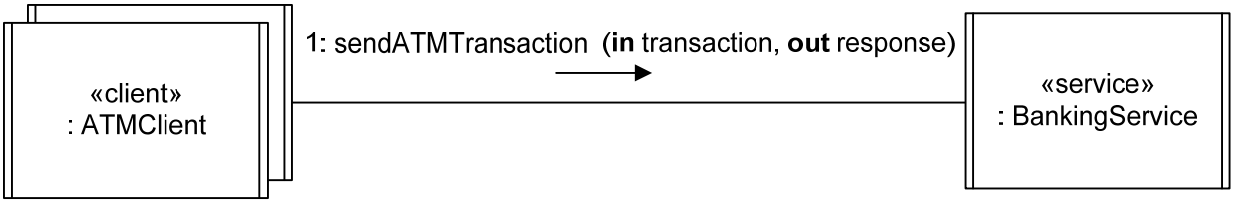


Figure 1: Client/Service pattern

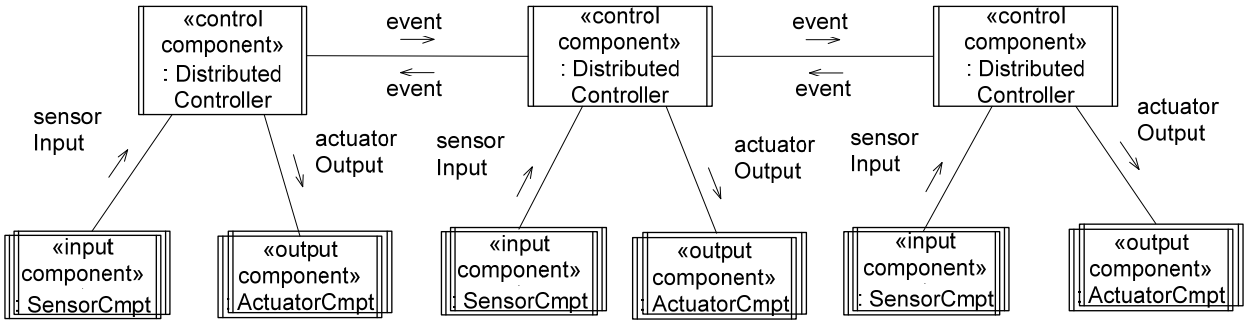


Figure 2: Distributed Control pattern

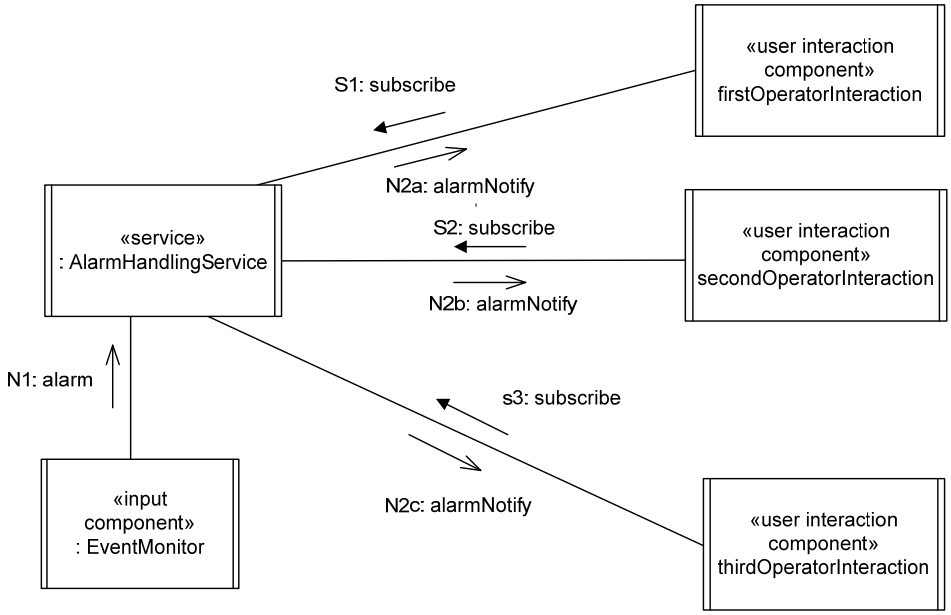


Figure 3: Subscription/Notification pattern

 ISBN: 978-1-61208-158-8

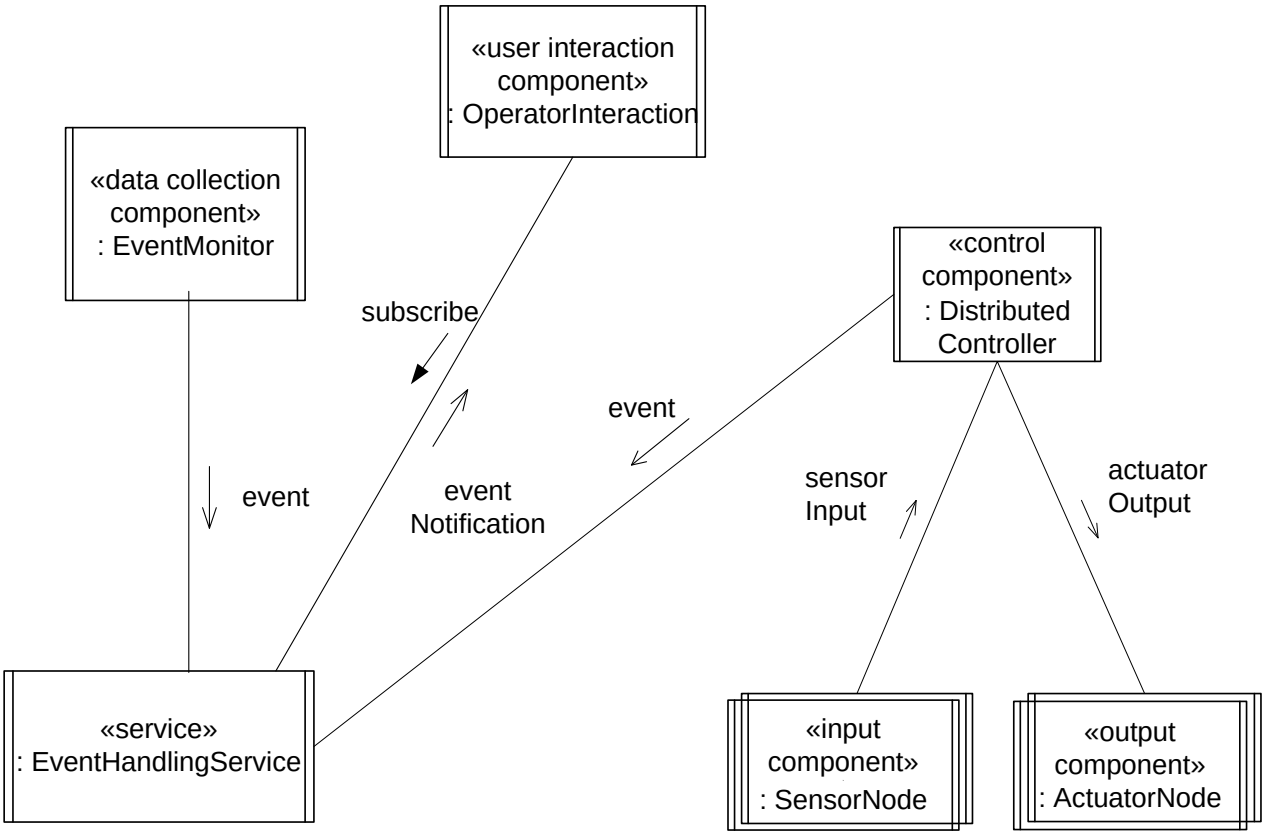


Figure 4: Emergency Monitoring and Control System

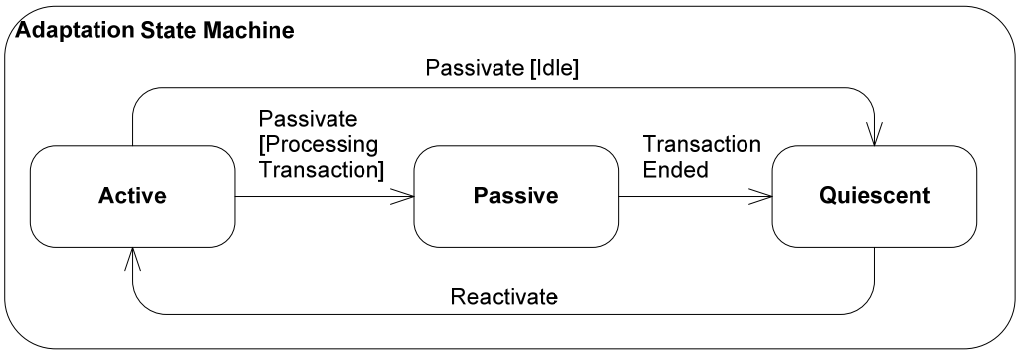


Figure 5: Software Adaptation State Machine

Issues of Persistence Service Integration in Enterprise Systems

Ádám Zoltán Végh, Vilmos Szűcs, Miklós Kasza, Vilmos Bilicki
Department of Software Engineering
University of Szeged
Szeged, Hungary
{azvegh,vilo,kaszam,bilickiv}@inf.u-szeged.hu

Abstract—**The increasing spread of smart sensors and multi-functional mobile devices extends the problem space of the integration issues appearing in information systems. The method of integrating different data sources (data providers, sensor devices, data hubs, etc.) highly affects system performance but development performance has to be considered, as well. One major topic where automation can help in both areas is persistence. A well-designed persistence layer service can be used by a diverse set of various enterprise applications. The application of the Service Oriented Architecture (SOA) paradigm can help in engineering such systems. Some methods available in a SOA-based system approach the problem at high level. This paper describes a well-maintainable solution at low level, on data model and data access levels. The main challenge this paper addresses is: how to increase the efficiency of integration in Java Enterprise Edition systems in cases when the data model and the interfaces of data access layer change frequently during development and even in maintenance phases. Based on real-life experience gathered during the execution of several telemedicine projects, our paper presents a solution for publishing a data model in the form of transferable objects where the developers do not have to care about the implementation of the assemblers dealing with transferable objects. The benefits and drawbacks have been identified with regards to performance and maintenance costs.**

Keywords-integration; persistence; data access; serialization; maintenance; code generation; Hibernate

I. INTRODUCTION

The heterogeneity of information systems leads to complex integration processes involving different applications and services. A major portion of integration problems cover domain model integration. When the domain entities change frequently, integration costs can increase significantly. This variability is definitely typical in the field of telemedicine, where a diverse set of sensors, data sources and systems have to work together. This increases the need of a cost-effective and sustainable model that supports a wide range of services and devices.

There are several well-known architectural patterns that help with building simple web applications. Specifically, a common solution, the *Data Access Object* (DAO) pattern is used often for decoupling persistence providers from business logic. However, it can be hard to define a really usable, separated persistence service that can be utilized by several other services – especially in the case of complex enterprise systems providing integrated services.

One of the popular enterprise platforms, the Java EE [1][2][3] platform enables the access of databases via Java Persistence API (JPA [4]) implementations. A typical JPA implementation serves as an object-relational persistence mechanism. It also has to utilize proxies that enable the lazy loading of referenced objects and collections (those not loaded along with the entities on the owning side of these relationships). The main problem of this object-relational persistence mechanism is that the objects managed by the JPA are not serializable. Neither are the container proxies (e.g. proxies in Hibernate [5]). Furthermore, these classes could not be published in any other way either. The proxies describe typical associations, aggregations, and inheritance, which is totally useless information after publishing an object, or these relations could not be resolved in a default way. The utilization *Data Transfer Objects* (DTOs [6][7]) can solve these problems.

A major disadvantage of applying a DTO-based approach is that it needs manually implemented transformations between the DTOs and the managed or entity objects. The implementation of such transformations includes a lot of repetitive tasks and thus infers the possibility of human errors and as such, productivity reduction. During the development of a real commercial system the following question emerges: how to solve the DTO-related development problems smarter and faster in a more maintainable way? Hibernate is widely used in persistence layers, but it does not include any acceptable toolkits that support proxy serialization.

An additional issue concerns integration middleware technologies. As almost every single modern enterprise system employs a middleware product at its heart, these technologies also affect the maintainability of an integrated system. Developers usually have to deal with middleware-related communication rules and interfaces regardless of the underlying transport mechanism, should it be an Enterprise Service Bus (ESB) or a web service-based solution (or anything else dealing with distribution issues). For example, in the case of an ESB-based system, each invocation is an assembled ESB message, published through the bus. The integration solution presented in this paper results in a model

 ISBN: 978-1-61208-158-8

addressing the aforementioned issues and is based on design patterns. Metrics-based evaluation (described later in the paper) points out the productivity and maintenance-related benefits of the solution.

The paper is organized as follows: Section II gives a detailed overview of the integration issues in the case of a typical persistence layer in an enterprise system. Section III describes the designed model and extensions for an integrated system. In Section IV a short example of data flow and transformations is introduced. This covers the automated persistence layer access. Section V aims at presenting the comparison and evaluation of the presented model and an already existing DTO-based solution. Finally, Section VI contains a summary of the results and a conclusion on the solution presented in this paper.

II. PERSISTENCE LAYER INTEGRATION ISSUES

A major advantage of Hibernate is the availability of a lazy initialization mechanism [8]. Using this type of object-collection handling, developers can save valuable time, typically for run-time. Hibernate proxies allow late binding during data access, i.e., the proxies are not resolved unless the appropriate accessor (getter) methods are called, and thus the time while the query would run is saved. As soon as the content of a managed object's collection gets needed for some operations, the proxy cares about running the proper query, de-serializes the results and returns them as a managed collection. However, a disadvantage appears when the managed object is serialized, since the proxies cannot be resolved. Traditional Hibernate proxies are not serializable. After serialization and de-serialization of entities, the proxies are trimmed and the lazy associations and aggregations are ignored. Afterwards, when these trimmed entities are returned to the persistence layer, Hibernate detects the ignored associations, and assumes these aggregations as deleted references: if cascading is set, the connected objects are removed and SQL DELETE commands are performed. Figure 1 explains the life cycle of object management. Integration can work only in the *Detached* state.

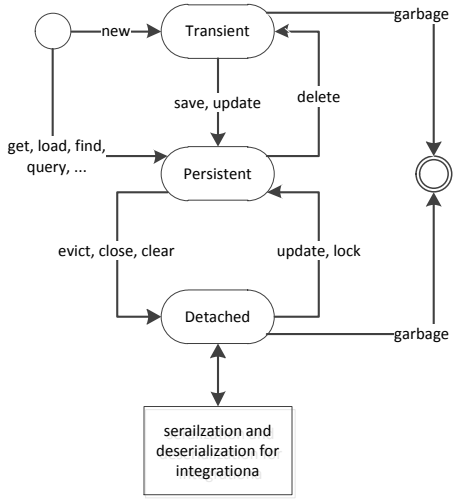


Figure 1. Life cycle of a managed object

In an integrated enterprise system, the DTO design pattern is applied widely in combination with Data Access Objects (DAOs) to overcome the issues regarding proxy serialization. The DTO/DAO model allows the sending and receiving of unmanaged objects and the serialization and de-serialization of them. The DAO-based data access layer often follows the System Façade design pattern. The conversion between the managed objects and the DTOs is provided by DTO assemblers. The DTO assemblers may follow various design patterns. DTO definitions are placed in separate classes. A key problem of the traditional solution is the cost of maintenance. When a DTO structure changes during development or maintenance (e.g. changes caused by third-party developers), maintenance costs increase. A major reason of this increase is that modifications are propagated across code.

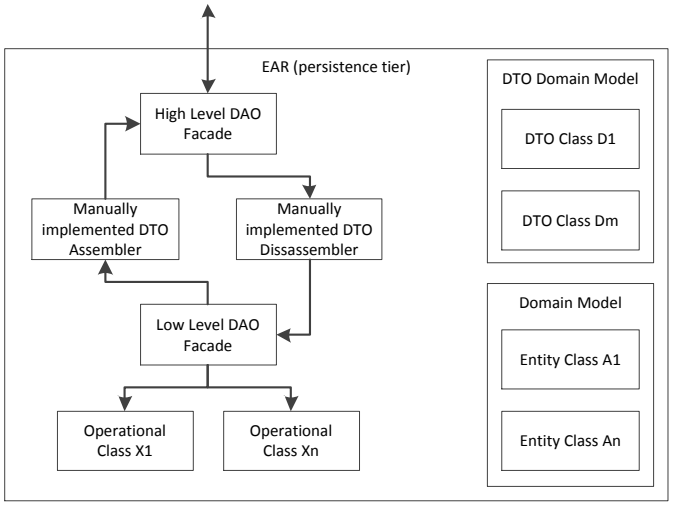


Figure 2. A classical type of DTO model

Data Transfer Objects are accessed by external systems through the DAO layer. Figure 2. explains the DTO transformation and the DAO: the DAO layer is hidden behind a System Façade, but this is not crucial in the model. In enterprise systems, the data provider tier (often the persistence tier) can publish DAOs even through a JNDI directory, or via any other suitable mechanisms. The client or the integration layer is usually responsible for using the DAO, receiving the DTOs, updating and sending them back to the data provider layer. As it was mentioned above, on the consumer side developers also have to deal with the DAO access mechanism at each invocation targeted at the data access layer. This mechanism is based on the integration middleware applied in the system.

III. ARCHITECTURE DESIGN

In this section, our solution is presented that allowed us to publish DTOs to the client side via DAOs in a way where the DAO interface definitions are the same as on the data provider side. Moreover, the DAO implementation automatically creates a huge part of code for the data consumer side (the client side). The presented model hides the different integration layers like ESB, which serialize the objects in

 ISBN: 978-1-61208-158-8

some manner. The automatic DTO conversion is not our own development, however the client side DAO generation and the whole system integration presents a new way.

One of the aforementioned problems is the DTO assembling and disassembling. In most cases, the assembler and disassembler codes are written manually and thus the DTO and the entity stored by JPA or Hibernate are nearly the same. Writing the assemblers and disassemblers in such a repetitive way causes code that is hard to maintain. Our methodology and architectural solution uses the Gilead framework to solve this maintenance and reimplementation task. The DTO transformation is executed at the data provider side where the Hibernate proxies are converted to Gilead proxies. This transformation solves the lazy proxy problem because the Gilead proxies are serializable. After the objects are detached, sent and returned, the proxies are restored by Gilead. Finally the returned objects are merged to the managed object instance. Figure 3. represents the high level layout of the Gilead based system.

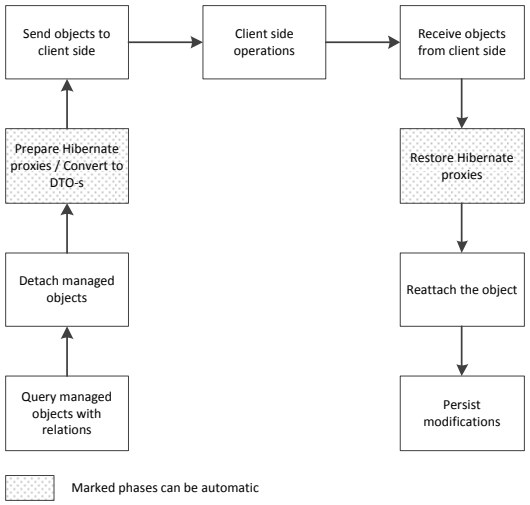


Figure 3. Lifecycle of a call

The mechanism of proxy replacement seemed to be simple: the use of the existing Gilead framework provided a solution for the problem. This enabled us to reduce the costs of assemblers and dissemblers written manually by developers where errors occurred very frequently due to careless modifications and redundant work. The other problem, the production of automatized DAOs, seems to be more complex. For example, a data provider layer can publish DAO interfaces through remote procedure calls or web services, but the clients in these cases have to resolve and call the appropriate DAO method manually and they need manual maintenance as well. As the DAO interfaces are available on the data consumer side regardless of the actual type of remote invocation mechanism used, they can be applied to an automatic service invocation delegation approach. We applied the following model for the DAO interface implementation on the client side.

A module based on the Abstract Factory design pattern is responsible for producing DAOs and for caching them. The DAO production is carried out with the help of Javassist [9]. The essence of generation is a method expecting a DAO interface and providing a DAO instance in return. It explores the DAO interface with the help of the Java Reflection API, and then – using the Javassist API and the underlying runtime build technique – it creates the appropriate Java code on the client side. Afterwards, the Java code is compiled and a new instance is returned. The factory is responsible for storing this instance in a cache in order to avoid unnecessary reconversion, since recompilation appears to be a time consuming process. This way, it should be performed only once at the server startup. The developed solution has only been tested with stateless DAOs like most of the solutions used in the integration (see Figure 4.).

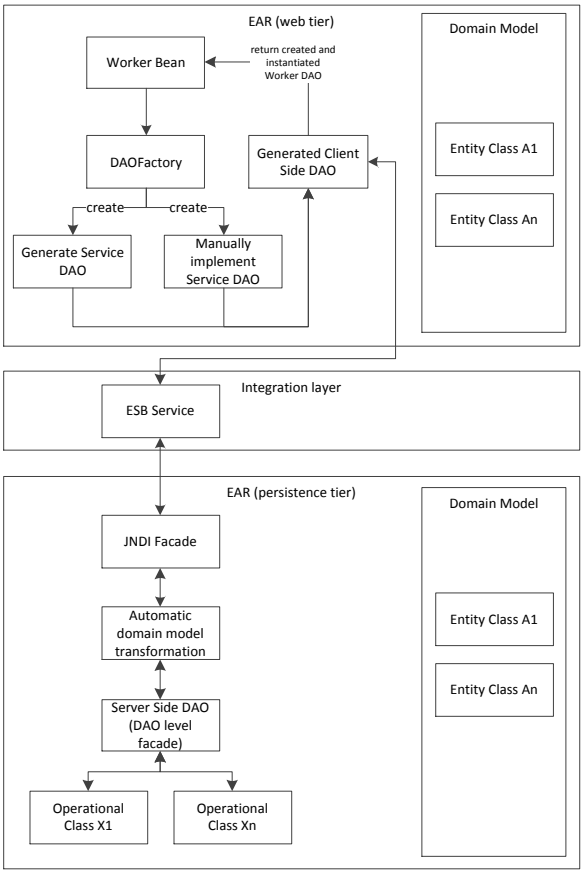


Figure 4. Architecture

By using this approach, we managed to reduce the redundant reimplementation of interfaces on the client side significantly and also to detect human errors occurring during development. The implementation we presented is applied in two real telemedicine projects: the mid-term DEAK-Medistance [10] and the long-term ProSeniis [11] projects. Both projects aim at sensor integration, data collection and storage. The DEAK-Medistance project has approximately 50 user screens, 65k lines of code and 41 entities. In the ProSeniis project there are approximately 150 user screens, 170k lines of code and 155 entities. Basically, the implementation of both projects can be divided into two parts: first, the solution for

 ISBN: 978-1-61208-158-8

resolving classical unserializable proxies on the data provider side has been detected; second, we succeeded in the automatization of DAO implementation production on the caller (client) side using DAO interfaces.

IV. FLOW OF AN INVOCATION

Considering a separated persistence service in an ESB-based, integrated enterprise system, this persistence service provides a data access layer via Enterprise Java Beans-based actual DAO instances registered in a JNDI directory provided by the integration middleware. When a data consumer service wants to interact with the persistence service, it asks the DAO factory for an appropriate DAO instance that implements a specific DAO interface. The factory inspects the methods defined by the interface with the help of the Java Reflection API and afterwards it constructs a Java class by extending the interface and its methods. Each method body assembles an ESB message with three parameters:

- The first parameter defines the *name of the concrete enterprise DAO bean*, which can be automatically prepared from the name of the interface.
- The second parameter defines the *name of the method that will be invoked*.
- Finally, the third parameter holds the *parameters of the invocation*.

As the factory returns the generated DAO instance, the data consumer service can simply invoke any of the methods. The body of the invoked method builds the ESB message and passes it to the bus.

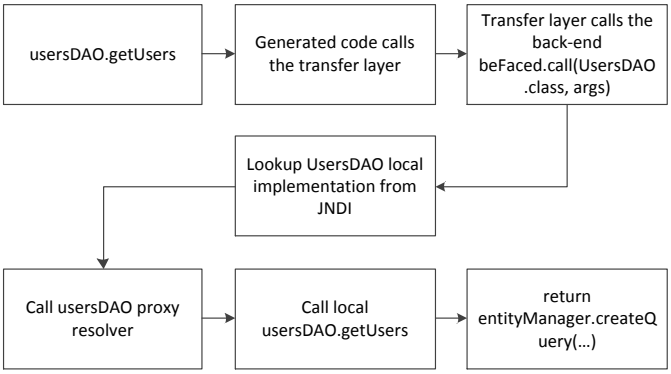


Figure 5. Flow of an invocation

The bus then transmits the ESB message to the persistence service, where a Session Façade interface is responsible for receiving all incoming messages. The receiver looks up the requested DAO (defined by the first parameter) in its JNDI directory. Afterwards, it gets a reference for the concrete DAO instance and then it tries to find the target method based on the second and third parameters of the incoming message using reflection again. If the corresponding method is found, the invocation is delegated to the DAO by passing the parameters defined in the third argument of the ESB message. The DAO uses the Gilead framework to clone and merge proxies. The result is returned in the same way via the bus (Figure 5).

As it can be seen, our approach simplifies the interaction between data consumer and data provider services in the system. Moreover, it hides the integration middleware-specific invocation delegation and thus developers dealing with the client side code can concentrate on the business logic – similarly to the method they use while applying the DAO objects in a simple 3-tier application. This integration model can be adapted for other types of interactions; it is not limited to persistence services.

V. EVALUATION OF PERFORMANCE AND MAINTAINABILITY

Performance and maintainability cannot be defined in a general way. This fact is true from another aspect as well, as during system integration a solution that is well maintainable, general and includes a lot of automatization can mean several limitations regarding speed and usability. Furthermore, the model that performed well in theory and in limited-scale experiments was subject to vertical profiling to make the measurement of the suspected performance decrease possible and to be able to compare the performance decrease with the maintenance advantages.

The experiment was performed on a developer workstation (Intel Core I5 CPU, 8GB RAM, Java 1.6.0_18). During the experiment, manual and automatic tests were administered. Logging was carried out with the Test & Performance Tools Platform (TPTP) 4.7 profiler; the platform was a JBoss 4.2.2.GA application server instance. TPTP ran as follows: the TPTP agent ran individually and remotely connected to a starting JBoss. The monitoring results were visualized with the help of Eclipse Helios Performance plugin, which was connected to the separately running TPTP agent.

A. Comparsion of performance

During performance measurement metrics were primarily assigned to running time. The consumer-producer call time and the time spent in the generated code were also measured. A real telemedicine application of the classical DAO and the latest re-written version of the application which showed the potentials of novelty presented system were compared.

TABLE I. PERFORMANCE METRICS (SMALLER VALUES ARE BETTER)

Metric	Classic DTO model	Presented model
CGT	0 ms	727 ms
TCMC	51 ms	56 ms
BTMC	3 ms	10 ms
CCS	15	18

The measured values are presented in Table 1. The following metrics were used to measure the absolute performance in a discrete way:

- CGT: *Average Time of Code Generation* per class – the time while the code generation runs. Since the presented model requires the automatic generation of

 ISBN: 978-1-61208-158-8

proxy classes, this is an additional one-time cost of the model. This metric is measured in milliseconds.

- TCMC: *Total Cost of Method Call* on client side. Average total time spent with calling methods provided by the service layer and called by the client. The time spent in the service methods is *also* included. This metric is measured in milliseconds.
- BTMC: *Base Time of assembling Method Call*. Average total time spent with calling methods provided by the service layer and called by the client. The time spent in the service methods is *not* included. This metric is measured in milliseconds.
- CCS: *Size of the Call Stack*. Total size of the stack on the client while calling a service method. This metric is measured in stack size.

It can be seen that our solution impacts performance in a negative way. However, this impact is quite low (some milliseconds), considering a simple call.

B. Comparsion of maintainability

The notion of maintainability can be divided into two parts: understandability and modifiability. As a rule of thumb, these characteristics are usually measured by implementing static metrics such as NC (Number of Classes), NA (Number of Attributes), DIT (Depth of Inheritance Tree), etc. Besides these product metrics it is advisable to measure process metrics as well. Process measurement involves simple metrics, such as the number of code lines that should be changed to provide a specific maintenance operation.

The following metrics were used for measuring maintainability:

- ISCL: *Interface Signature Change by Lines of code*. This metric measures the average number of code lines that need to be changed after modifying the signature of an interface method.
- AIL: *Addition of new DAO interface*. This metric measures the number of code lines that need to be changed after the introduction of a new DAO interface.
- DACL: *Change of a domain class attribute set*. This metric measures the average number of code lines that need to be changed after modifying the attribute set of a domain entity.
- DCML: *Modified lines when adding a new class in the domain model*. This metric measures the number of code lines that need to be changed when adding a new class in the domain model (excluding the class definition itself).
- DACC: *Change of the domain class Attribute set*. This metric measures the number of positions in code that need to be changed after modifying the attribute set of a DAO class (including the propagation of the changes).
- DCMC: Count of modifications when adding a new class in the domain model (except the class definition itself). This metric measure the number of positions in code that need to be changed after the introduction of a new container entity.

TABLE II. MAINTENANCE METRICS (SMALLER VALUES ARE BETTER)

Metric	Classic DTO model	Presented model
ISCL	8	5
AIL	8	4
DACL	18	6
DCML	approx. 2x class size	0
DACC	4	1
DCMC	3	0

As it can be seen in Table II, the measurement revealed that since code developers does not have to deal with DTO codebase maintenance, significant time can be saved. This is true only in cases when the DTO matches a managed object definition or its subset.

VI. CONCLUSION

The generation of consumer-side DAO implementation significantly reduces development and maintenance costs. The presented solution enables various types of simplified service invocation delegation in the integration layer. If we want to enable a new integration layer, our only task is to implement the specific generator and the connector at the data producer side. We can also state that automated DTO transformation means some decrease in performance. Tasks done by Gilead proved to be the bottleneck. As a future development, using own assemblers and disassemblers instead of Gilead could be a solution. Regarding maintainability, the code structure and costs of maintenance have improved to a big extent. Less maintenance of code means fewer human errors, which is critical in rapid development cycles. Saving the development costs of the DTO assemblers can mean lower profit. A bigger profit occurs in the reusability of the DTOs on the client side. The generators of the designed DAOs can be reused and extended for any arbitrary serial integrated layer.

Employing DTOs for divided domain model, the managed entities can pose certain limitations. The current solution can be used successfully only in cases where there is no need to employ DTOs with aggregated data.

ACKNOWLEDGMENT

The work presented in this paper was partly funded by the National Innovation Office, Hungary (project No. OM-00191/2008-AALAMSRK ‘ProSeniis’), Telenor 19/55 1I066, Nokia Komarom 19/55 1C117 and GOP-1.1.2-07/1-2008-0007.

REFERENCES

[1] Beth Stearns, Inderjeet Singh, and Mark Johnson, *Designing enterprise applications with the J2EE platform*, 2nd ed.

 ISBN: 978-1-61208-158-8

[2] "Java 2 Platform, Enterprise Edition (J2EE) Overview." [Online]. Available: http://java.sun.com/j2ee/overview.html. [Accessed: 31-Mar-2011].
[3] Adam Bien, *Real World Java EE Patterns Rethinking Best Practices*. 2009.
[4] Merrick Schincariol, and Michael Keith, *Pro JPA 2: Mastering the Java(TM) Persistence API (Expert's Voice in Java Technology)*. 2009.
[5] Christian Bauer, and Gavin King, *Hibernate in Action*. Manning Publications Co.
[6] Alexandar Pantaleev, and Atamas Rountev, "Identifying data transfer objects in EJB applications," in *Proceedings of the 5th International Workshop on Dynamic Analysis*, 2007, p. 5.
[7] Martin Fowler, *Analysis Patterns: Reusable Object Models*. Addison-Wesley Professional.
[8] "Chapter 19. Improving performance." [Online]. Available: http://docs.jboss.org/hibernate/core/3.3/reference/en/html/performance.html. [Accessed: 31-Mar-2011].
[9] Shigeru Chiba, "Javassist - A Reection-based Programming Wizard for Java," *Proceedings of OOPSLA'98*, 1998.
[10] "Főoldal | medistance.hu HU." [Online]. Available: http://medistance.hu/. [Accessed: 31-Mar-2011].
[11] "ProSeniis." [Online]. Available: http://www.proseniis.hu. [Accessed: 31-Mar-2011].

 ISBN: 978-1-61208-158-8

Case Study Towards Implementation of Pure Aspect-oriented Factory Method Design Pattern

Žilvinas Vaira
Vilnius University
Institute of Mathematics and Informatics
Akademijos 4, LT-01108 Vilnius, Lithuania
zilvinas.vaira@ik.ku.lt

Albertas Čaplinskas
Vilnius University
Institute of Mathematics and Informatics
Akademijos 4, LT-01108 Vilnius, Lithuania
albertas.caplinskas@mii.vu.lt

Abstract—**The paper investigates a case of application of Factory Method design pattern in the context of aspect-oriented design. The case encompasses whole path of design pattern transformation from object-oriented to aspect-oriented design and its application to a particular design context. The main assumption is that there exist design patterns that solve design problems in a similar way for both programming paradigms and that such design patterns can be expressed in the terms of a corresponding programming paradigm. The paper uses design pattern classification and design pattern transformation technique proposing that 20 of Gang of Four 23 design patterns can solve design problems in a similar way to both Object-Oriented Programming and Aspect-Oriented Programming and can be successfully adapted for the needs of aspect-oriented design. The research presents a detailed explanation and examples how to apply proposed technique and discusses elaborated results.**

Keywords - Aspect-Oriented Programming; Object-Oriented Programming; Design Patterns; Factory Method; Framework design.

I. INTRODUCTION

The main intent of this paper is to present an exemplary case showing how object-oriented (OO) design patterns can be redesigned into pure aspect-oriented (AO) design patterns. The complete description of the theoretical discussion, design pattern classification and detailed pattern redesign technique description is proposed in [19]. The research is concentrated on Gang of Four (GoF) 23 [4] design patterns and investigates the question of the possible similarities between two programming paradigms – object-oriented programming (OOP) and aspect-oriented programming (AOP) [11]. As a result, it states that 20 of GoF 23 design patterns can be adapted to solve problems of aspect design. The main contribution of this paper is the experimental evaluation of the [19] proposed redesign technique. It is performed by detailed analysis of redesign technique application to a real life system design. Vaira and Čaplinskas [19] provided theoretical reasoning and models of the redesigned patterns. However, it does not give any insight of practical application of the technique except some hypothetical application context. The results of this paper provide strong evidence in the form of implementation diagrams and detailed description that such design patterns are applicable to real life systems design. It can be stated as a qualitative experimental evaluation of the previous theoretical research. To perform this evaluation, the Factory Method design pattern has been chosen for this research. The case of Factory Method design pattern can be treated as a critical case [16] because it corresponds to the creational design patterns, which are less to be likely acceptable for redesigning into aspects, because they are highly related to creation of objects. Creation of aspects is far different from creation of objects, because aspects are singletons by their nature and its creation in most AO language implementations is handled by aspect weaver automatically. Hence, this paper presents strong evidence that even creational OO design patterns can be adapted to design AO ones.

The major part of the paper includes details of the OO Factory Method design pattern redesign process into AO Factory Method pattern and investigates its application in the context of AO framework design. The main questions that we aim to answer in this paper include: are such AO design patterns applicable in real life applications and does AO representation of Factory Method design pattern change its purpose anyhow? All examples are presented using (Unified Modeling Language) UML class diagrams and stereotyped class diagrams for aspects. The resulted applications are implemented using Java and AspectJ [12] programming languages.

The remaining part of this paper is organized as follows. Section 2 describes the process of Factory Method pattern redesign. Section 3 demonstrates the applicability of pure AO Factory Method design pattern for designing an AO framework. Section 4 analyses related works. Section 5 presents a discussion. Finally, Section 6 concludes the paper.

II. REDESIGNING FACTORY METHOD PATTERN FOR ASPECTS

In this paper, we use terms “redesign” and “pure AO design pattern” in order to distinguish the technique from other design pattern transformation techniques proposed in [6], [8], [9]. By redesign, we mean that design patterns must be reworked from the perspective of its design problem and solution, not by performing simple refactorings or other transformations. Transformation techniques proposed in [8],

 ISBN: 978-1-61208-158-8

[6], [9] search alternative solutions for the same design problem. Our redesign technique redefines design problem for aspects and searches for a design solution using aspects only. More detailed comparison of these techniques could be found in Section IV. The proposed redesign technique is concentrated on two paradigms only, namely OOP and AOP. Such paired paradigm use generates new types of design structures that involve concepts and relations from both paradigms. Moreover, it forces AO paradigm language implementations, such as AspectJ, to inherit elements of a larger scale base paradigm, on which it is built up. Resulted AspectJ language implementation still includes other small scale paradigm elements that are introduced by AOP [20]. This results in complex structures that are problematic to be developed.

The main idea is that some design problems may be stated as common to both paradigms and others as specific with regard to paradigms analyzed (i.e. OOP and AOP). In this case the solution of design pattern that solves these design problems could be performed on both paradigms involving specific paradigm constructs only.

A. Redesign technique

The redesign is based on statement alleging that when OO design pattern can be implemented in AspectJ by using AO constructs only, it can be considered as a pattern that solves similar design problem. It seems that in such case both OO and AO patterns solve the same design problem, but their applicability differs. Thus, the problem in some sense is also different: the OO pattern solves a design problem for objects, whereas the AO pattern solves it for aspects.

Redesign is based on the similarities between aspects and classes, despite the fact that they are different concepts:

- Aspects, similarly as classes, can define data members and behaviors for crosscutting concerns [14]. They can also be defined as abstract entities, or implement Java interfaces.
- Aspects can be used as collaborative entities and build inheritance hierarchies in similar way as it is done with classes.

However, one of the main differences is the fact that aspects cannot be directly instantiated. There is a possibility to use several instances of one aspect in a program by declaring an aspect per object or per control flow in a program. In such case the instantiation still differs from the one that is done with classes. For this reason we refer to aspects as singletons. Redesign technique involves 3 main steps (see Figure 1):

- If a GoF 23 pattern can be implemented using singletons only, it is regarded as a candidate to be a design pattern for rewriting to AspectJ.
- All classes in the candidate pattern should be replaced by aspects.
- The candidate pattern should be analyzed in order to discover and remove irrelevant data members and methods.

These steps are generalized from the original. More detailed technique description can be found in [19].

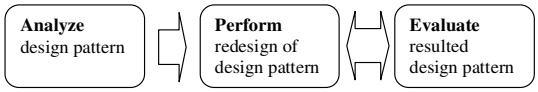


Figure 2. Redesign technique

B. Redesign description

In the case of Factory Method design pattern we are dealing with, it may seem that the AO solution has no sense, because Factory Method belongs to creational pattern category and is highly related to creation of objects. In the AO paradigm we in most cases deal with the singletons only and in fact the creation of aspects cannot be managed directly by other aspects. However, it does not mean that the redesign technique can not be performed on Factory Method design pattern. The creation of aspects can be replaced by passing a reference to already created aspect. In order to do this we can use *AspectOf* method instead of constructor method. *AspectOf* corresponds to an analogue *InstanceOf* that is used for referencing singletons. We will demonstrate that AO solution of Factory Method can be redesigned using

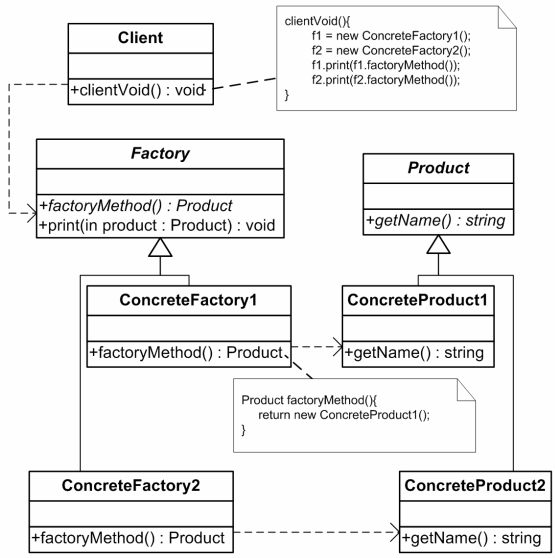


Figure 1. Factory Method design pattern (OO solution)

proposed technique.

The first step is to perform analysis of the pattern to inspect if it can be regarded as candidate for rewriting. The Factory Method design pattern defines an abstract method that can be overridden by subclasses for creating objects that belong to different classes [4]. There are several other variations of the pattern (e.g. the parameterized factory method), but in this particular case we use the general one. The main elements of the general case of Factory Method (see Figure 2) design pattern are:

- *Factory*, an abstract class that contains abstract operation *factoryMethod*, which is overridden by its subclasses,

 ISBN: 978-1-61208-158-8

- *ConcreteFactory1* and *ConcreteFactory2*, concrete *Factory* classes overriding *factoryMethod*, which creates and returns the object of *ConcreteProduct1* or *ConcreteProduct2* respectively.
- *Product*, an abstract class that contains the abstract operation *getName* and defines the interface of *Product* type objects,
- *ConcreteProduct1* and *ConcreteProduct2*, concrete *Product* classes that implement the *getName* operation using some concrete method, and
- *Client*, the class that invokes the *factoryMethod* of the *Factory* object.

There is no critical reason indicating that Factory Method design pattern can not be implemented using singletons only. Abstract classes can be replaced by abstract aspects, subclasses by specializing aspects. The constructors of *ConcreteProduct1* and *ConcreteProduct2* can be replaced by *AspectOf*. All other operations remain the same as in classes.

When it is decided that the Factory Method is a candidate for redesigning, the second step can be performed in Figure 3. The resulted AO Factory Method solution helps to get a

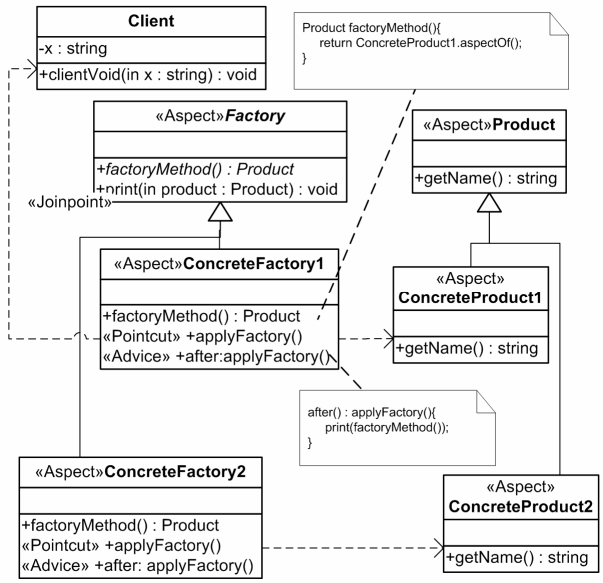


Figure 3. Application of the AO Factory Method design pattern

reference to the necessary aspect defined by specialized *Factory* aspect. This is an analogous solution to that of OO Factory Method design pattern. The main difference is that instances of aspects are created only once and each time we execute *factoryMethod* particular *Product* instance is passed as an argument.

The last step of evaluation of resulted pattern involves possible refactorings to enhance the resulted design and to test its applicability. The main variation of the pattern can be performed by changing or adding pointcuts and advice. The current model includes pointcuts and advice in subaspects of *Factory* aspect and in this way it is defined when *factoryMethod* operation is invoked. Another place for defining pointcuts and advice could be subaspects of *Product* aspect. More comprehensive designs of pattern behavior could be resulted by predefining some pointcus or advice in abstract aspects. The important difference of AO design pattern comparing to its OO analogue is that the developer is limited with a number of predefined subaspects it can use at the same time (except of above mentioned cases per object or per control flow aspects). However, it does not change the principal behavior of this design pattern and demonstrates that AO design pattern preserves all essential elements of the OO pattern.

An example of the application of the AO Factory Method pattern is analyzed in the following part of the paper. In this example, we are dealing with the complex logging concern in a simulation domain framework.

III. APPLICATION OF PURE AO FACTORY METHOD

SimJ simulation framework is used as experimental application providing necessary context for implementing AO Factory Method design pattern. The main research interest is concentrated on logging concern, which has a crosscutting issues that need to be eliminated and the feature of logging needs to be made customizable. SimJ is a simulation framework used for developing simulation applications based on discrete events.

The logging concern in a framework suffers from crosscutting. Pieces of the code belonging to it are scattered and tangled through the remaining part of a framework. The complexity of a logging functionality of this framework makes it a sufficient candidate to apply the AO Factory Method design pattern presented in Figure 3. The framework has several different kinds of things to be logged and must remain customizable in a concrete specialization of a framework. The current version of the framework allows customizing logging. However, it is handled beyond the bounds of logging concern individually by every entity that needs to be logged. The main purpose of application of AOP is to exclude all pieces of code related to logging concern and combine them in aspects. Although the design of these aspects is not an ordinary task to complete, design pattern could be applied to handle it.

The AO Factory Method design pattern was introduced to deal with the following issues: different logging behavior for resources and several kinds of events was necessary and the complexity of triggering of this behavior required its separation. Different behavior of logging was modeled using product hierarchy in Factory Method pattern. The triggering structure of logging behavior was designed using hierarchy of factories Figure 3. The resulted implementation of logging concern is presented in Figure 4. The UML diagram contains complete design that includes two additional instances of Template Method (design pattern is usually used in composition with factories). The stereotype “Hook” is used to denote customizable framework methods in aspects.

Consequently, several advantages can be noticed: all the logging functionality and related code is localized in one place and the customization of the logging concern can be carried out separately from the rest of the hot spots. This also

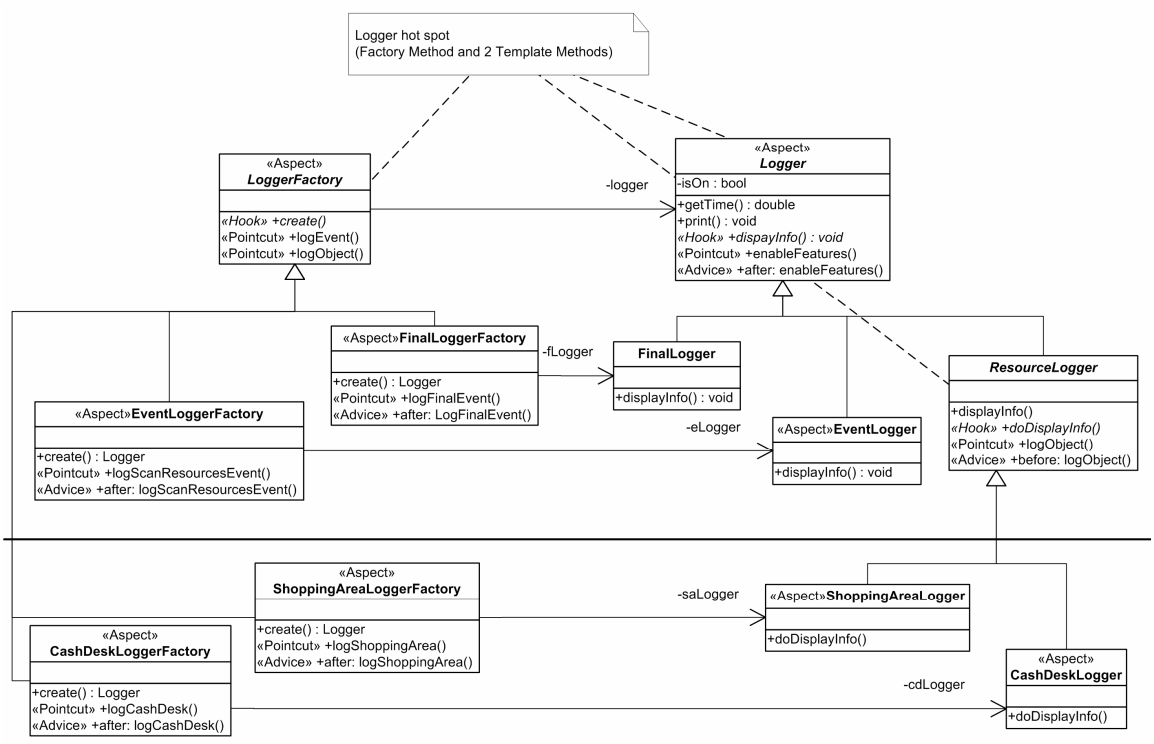


Figure 4. Factory Method desgin pattern (AO solution)

means that maintenance and unplug ability features of the logging were increased. This implementation allows flexible customization so that logging of events and resources can be done separately and the joinpoints triggering logging behaviors can be customized independently. A high number of aspects can be considered as a shortcoming. This is probably related to the complexity of the logging concern behavior.

IV. RELATED WORKS

OO design patterns can be redefined for the AO paradigm in several different ways. Implementation of the OO design pattern in Java, can be directly replaced by the analogous code written in AspectJ [6], [8] a native AO solution can be introduced to the same problems that are addressed by the OO design pattern [9] or pattern solving AOP specific design problems can be elaborated [2], [7], [14], [15]. There is no concrete technique describing how to discover patterns solving AOP specific design problems. However, two different design pattern transformation techniques can be distinguished and compared to the one analyzed in this paper:

a) The authors of [6] and [8] use very similar pattern transformation technique. They introduce AOP constructs to deal with the problems related to crosscutting in the pattern solution. The design problem solved by the pattern does not change and the main idea of the solution remains the same.

b) Authors of [9] use slightly different technique. The main idea is that design solution still must deal with the same problem. However, aspects are used to search for an alternative solution, different than the main idea provided by original design pattern. Both of these techniques provide solutions to the same design problems. Such solutions are alternatives and can be compared.

c) Our technique, presented in [19], is slightly different, because we redefine a design problem for aspects. Considering the similarities between AOP and OOP paradigms we say that a similar design problem that occurs when designing objects can also occur while designing aspects. In such a case we can use the same design idea that has solved the design problem for objects, but this time only aspects are used. In result, the design pattern achieved using this technique is not an alternative solution to the same design problem. It is more like a new AOP design pattern solving a similar to OOP design problem in a similar to OOP way.

A number of quantitative evaluations have attempted to measure the effectiveness of the implementation [5], [8] and [3]. As design patterns can be composed in many different ways and crosscut each other, most of these quantitative assessments agree that aspectization of patterns can significantly improve OO implementations. However, in some cases the results depend on the patterns involved, design complexity, and other circumstances as discussed in [3]. The main problems commonly reduced by the use of aspects are related to code scattering and code tangling. So, it

is reasonable to expect that implementations in AO languages will at least partly solve these problems.

A framework that is used to provide some contextual evidence for AO Factory Method application is considered as the software framework described in [10]. It states that application framework is a reusable, „semi-complete" application that can be specialized to produce custom applications. The application of pure AO design patterns produces a new kind of application framework that we refer to an AO application framework. Similar AO framework design, where aspects were used as glue code for gluing framework core and its application was presented in [17]. A more comprehensive and a more related to this paper AO framework design, that includes the use of customizable aspects, is presented by the following researches [1], [18], including more complex design structures that involve some idioms of AspectJ in [13].

V. DISCUSSION

There are several debatable issues that we would like to discuss. The main of them is the use of aspects as collaborative entities. The designs that include abstract aspect hierarchies hold references and invoke calls to other aspects help to create reusable and flexible implementation structures. These are the main features used to create collaborations of classes in OOP. However, such structures also increase the tangling of the implementation code, which is an issue that AOP used to deal with. It is not always clear, what the constraints of collaborations in aspects are and when a threat of creating too complex designs of aspects appears. We assume that collaborations of aspects are beneficial unless they overstep the boundaries of related concerns.

The Singleton nature of aspects is the second issue. Though, aspects in AspectJ are by default singletons, in special cases aspects can be also instantiated per object or per control flow. From this perspective it is still questionable whether aspects should be treated as singletons or not. Direct instantiation of aspects in AspectJ language is forbidden. Aspects can be globally referenced using static method *aspectOf* and it is not quite compatible with the direct creation. Another problem is that if it were allowed to create several instances of the same aspect at a time, the behavior advised by aspects might repeat several times or act in other unexpected ways. As a result, there may be difficulties related to aspect instantiation control. This is the main reason why we suggest following the Singleton nature of aspects and treating per object and per control flow aspects as special cases of singletons.

VI. CONCLUSIONS AND FUTURE WORK

The paper demonstrated that design patterns solving similar design problems in both, AOP and OOP paradigms, could be used to deal with crosscutting and to design customizable aspects in frameworks. The investigated case of Factory Method design pattern shows that even creational design patterns can be applied for this purpose. It promotes the elimination of crosscutting behavior and localization of scattered implementations. Moreover, this crosscutting behavior can be designed as a reusable hot spot in a framework and customized in a framework application. The purpose of Factory Method design pattern in AOP is slightly changed comparing to OOP. Instead of creating factories it only passes reference to the necessary aspect. In some cases the use of the pure AO design patterns only may be insufficient. They should be used in compositions with available design patterns from other categories of AO design. It is reasonable to expect that compositions with patterns for designing pointcuts and advice could increase the applicability of existing ones or even create new AO design patterns.

Further investigations of pure AO design pattern applications to design programs are necessary. The investigation towards other patterns solving similar design problems in other paradigms is also intended.

ACKNOWLEDGMENT

The authors wish to thank Software Engineering Research Group headed by Prof. Jacques Pasquier for providing SimJ framework for experimental application. Personal thanks to Prof. Jacques Pasquier, Dr. Patrik Fuhrer and Minh Tuan Nguyen for inspiring and initial guiding of related research.

REFERENCES

[1] P. Arpaia, M.L. Bernardi, G. Di Lucca, V. Inglese, and G. Spiezia, "Aspect Oriented-based Software Synchronization in Automatic Measurement Systems", In *Proceedings of Instrumentation and Measurement Technology Conference*, IMTC 2008, IEEE, pp. 1718 – 1721, 12-15 May 2008.

[2] M. Bynens and W. Joosen, "Towards a Pattern Language for Aspect-Based Design", In *Proceedings of the 1st workshop on Linking aspect technology and evolution* (PLATE '09), Charlottesville, Virginia, USA date:March 2 - 6, 2009. ACM, pp. 13-15.

[3] N. Cacho, E. Figueiredo, C. Sant´Anna, A. Garcia, T. Batista, and C. Lucena, "Aspect-oriented Composition of Design Patterns: a Quantitative Assessment", *Monografias em Ciência da Computação*, vol. 5, no. 34. Pontifícia Universidade Católica do Rio de Janeiro, Brasil, 2005.

[4] E. Gamma, R. Helm, R. Johnson, and J.Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Professional, 1994.

[5] A.Garcia, C.Sant'Anna, E. Figueiredo, and U. Kulesza, "Modularizing Design Patterns with Aspects: A Quantitative Study", In: *Proceedings of the International Conference on Aspect-Oriented Software Development* (AOSD'05), Chicago, USA, 14-18 March 2005. ACM Press, pp. 3-14.

[6] O. Hachani and D. Bardou, "Using Aspect-Oriented Programming for Design Patterns Implementation", In *Proceedings of 8th International Conference on OOIS 2002*, Position paper at the Workshop on Reuse in Object-Oriented Information Systems Design, Montpellier, France - Sept. 2-5 2002.

[7] S. Hanenberg and A. Schmidmeier, "Idioms for building software frameworks in AspectJ", In *Y. Coady, E. Eide, D. H. Lorenz (Eds.) Proceedings of the 2nd AOSD Workshop on Aspects, Components, and Patterns for Infrastructure Software* (ACP4IS), College of Computer and Information Science, Boston, Massachusetts, 2003, pp. 55-60.

[8] J. Hannemann and G. Kiczales, "Design pattern implementation in Java and AspectJ", In *Proceedings of the 17th Conference on Object-Oriented Programming, Systems,*

 ISBN: 978-1-61208-158-8

Languages, and Applications (OOPSLA '02), ACM Press, 2002, pp. 161-173.

[9] R.Hirschfeld, R. Lämmel, and M. Wagner, "Design Patterns and Aspects – Modular Designs with Seamless Run-Time Integration", In *Proceedings of the 3rd German Workshop on Aspect-Oriented Software Development* (AOSD-GI 2003), 2003, pp. 25–32.

[10] R. E. Johnson and B. Foote, "Designing Reusable Classes", *Journal of Object-Oriented Programming*, June/July 1988, vol. 1, no. 2, pp. 22-35.

[11] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. V. Lopes, J.-M. Loingtier, and J. Irwin, "Aspect oriented programming", In *Proceedings of European Conference on Object Oriented Programming, ECOOP*, 1997, vol. 1241, pp. 220–242.

[12] G. Kiczales, E. Hilsdale, J. Hugunin, M. Kersten, J. Palm, and W. G. Griswold, "Getting started with AspectJ", *Communication of the ACM*, October 2001, vol. 44, no. 10, pp. 59–65.

[13] U. Kulesza, V. Alves, A. Garcia, C. J. P. de Lucena, and P. Borba, "Improving Extensibility of Object-Oriented Frameworks with Aspect-Oriented Programming", In *Proceedings of Intl Conference on Software Reuse (ICSR)*, Torino, Italy, pp. 231-245, 2006.

[14] R. Laddad, *AspectJ in Action: practical aspect-oriented programming*, Manning Publications Co, 2003.

[15] R. Miles, *AspectJ Cookbook*, O'Reilly Media, 2004.

[16] Charles C. Ragin, ""Casing" and the process of social inquiry", In *Charles C. Ragin and Howard S. Becker (eds), What is a Case? Exploring the Foundations of Social Inquiry*, Cambridge: Cambridge University Press, 1992, pp. 217–26.

[17] A. Rausch, B. Rumpe, and L. Hoogendoorn, "Aspect-Oriented Framework Modeling", In *Proceedings of the 4th AOSD Modeling with UML Workshop*, UML Conference 2003, October 2003.

[18] A. L. Santos, A. Lopes, and K. Koskimies, "Framework specialization aspects", In *Proceedings of AOSD '07 the 6th international conference on Aspect-oriented software development*, ACM New York, NY, USA 2007, pp. 14 - 24.

[19] Ž. Vaira and A. Čaplinskas, „Paradigm-independent design problems, GoF 23 design patterns and aspect design", *Informatica*, Institute of Mathematics and Informatics, Vilnius, in press.

[20] V. Vranić, "AspectJ Paradigm Model: A Basis for Multi-Paradigm Design for AspectJ", In *Jan Bosch, editor, Proc. of the Third International Conference on Generative and Component-Based Software Engineering* (GCSE 2001), LNCS 2186, Erfurt, Germany, September 2001, pp. 48-57, Springer.

 ISBN: 978-1-61208-158-8

A Language for Modeling Patterns for Extra-Functional Requirements

Brahim Hamid
IRIT, University of Toulouse, France
118 Route de Narbonne, 31062 Toulouse Cedex 9
France
hamid@irit.fr

***Abstract*—Model-driven engineering is well suited for the development of safe and heterogeneous systems, as it enhances the separation of concerns (i.e security, performance, analysis, simulation, etc.) through a declarative specification of behavior, e.g., by means of models that describe a system's functioning (where there are predefined configurations). In this paper, we deal with the idea of using patterns to describe extra-functional concerns as recurring design problems in specific design contexts, and to present a well-proven generic scheme for their solutions. To achieve this goal, we propose a Pattern Modeling Language to get a common representation to specify patterns for several domains. Our proposition is based on several levels of abstraction, for instance generic design (domain independent) and specific design (domain specific) levels. The aim of the generic design level is to catch, at high level, a set of generic properties by determining in advance if the artifact (e.g., pattern) has or uses a certain kind of generic properties. Then, specific domain design level allows to make more dedicated information. The approach enables us to define an engineering approach based on a repository of models and practices. It ensures separation of engineering concerns and roles between (1) application experts, (2) concerns experts and (3) MDE experts. The advantage of the language is illustrated by the modeling of the authorization pattern.**

***Index Terms*—Multi-Concerns engineering, Extra-Functional Properties, Pattern, Meta-model, Model Driven Engineering.**

I. Introduction

Extra-functional concerns become a strong requirement as well as more difficult to achieve even in safety critical systems. They can be found in many application sectors such as automotive, aerospace, and home control. Such systems come with a large number of common characteristics, including real-time and temperature constraints, computational processing, power constraints and/or limited energy and common extra-functional: such as dependability, security as well as efficiency [12]. Domains dealing with these concerns covers a wide spectrum of applications ranging across embedded real time systems, commercial transaction systems, transportation systems and military space systems, to name a few. The supporting research includes system architecture, design techniques, validation, modeling, software reliability and real-time processing.

The integration of such concerns, for instance security, safety and dependability, requires the availability of both application development and concerns expertises at the same time. Many domains are not traditionally involved in this kind of issue and have to adapt their current processes. Typically, such requirements are developed ad-hoc for each system, preventing further reuse beyond such domain-specific boundaries.

Safety critical systems require a high level of safety and integrity. Therefore, the generation of such systems involves specific software building processes. These processes are often error-prone because they are not fully automated, even if some level of automatic code generation or even model driven engineering support is applied. Furthermore, many critical systems also have assurance requirements, ranging from very strong levels involving certification (e.g., DO178 and IEC-61508 for safety relevant embedded systems development) to lighter levels based on industry practices.

Over the last two decades, the need for a formally defined safety lifecycle process has emerged. The integration of extra-functional mechanisms is still new in many domains. Hence capturing and providing this expertise by the way of specific patterns can enhance safety critical systems development. Model-Driven Engineering (MDE) provides a very useful contribution for the design of these systems, since it bridges the gap between design issues and implementation concerns. It helps the designer to specify in a separate way extra-functional requirements at an even greater level that are very important to guide the implementation process. Of course, a MDE approach is not sufficient but offers an ideal development context. While using a MDE framework, it is possible to help concerns specialists in their task.

The question remains at which step of the development process to integrate these patterns. As a prerequisite work, we investigate the design process of patterns. The goal of the paper is to propose a new pattern development technique in order to make easy their use in a building process of software applications with multi-concerns support. Reaching this target requires to get (i) a common representation of patterns for several domains and (ii) a flexible structure for a pattern.

According to Buschmann [2], a pattern for software architecture describes a particular recurring design problem that arises in specific design contexts, and presents a well-proven generic scheme for its solution. Unfortunately, most of exiting patterns are expressed in a textual form, as informal indications on how to solve some security problems. Some of them use more precise representations based on UML [1] diagrams, but

 ISBN: 978-1-61208-158-8

these patterns do not include sufficient semantic descriptions in order to automate their processing and to extend their use. Furthermore, there is no guarantee of the correctness use of a pattern because the description does not consider the effects of interactions, adaptation and combination. This makes them not appropriate for automated processing within a tool-supported development process. Finally, due to manual pattern implementation use, the problem of incorrect implementation (the most important source of security problems) remains unsolved.

The solution envisaged here is based on meta-modeling techniques to represent patterns at a greater level of abstraction. Therefore, patterns can be stored in a repository and can be loaded in function of desired properties. As a result, patterns will be used as brick to build a applications through a model driven engineering approach.

The work is conducted in the context of a framework called *SEMCO* for System and software Engineering for embedded systems applications with Multi-COncerns support. We build on a theory and novel methods based on a repository of models which (1) promote engineering separation of concerns, (2) supports multi-concerns, (3) use *patterns* to embed solutions of engineering concerns and (3) supports multi-domain specific process. This project is three-folded: providing repository of modeling artifacts, tools to manage these artifacts, and guidelines to build complete engineering systems.

The rest of this paper is organized as follows. An overview of our approach is presented in Section II. Then, Section III describes in detail the pattern modeling language we propose. Section IV presents in depth the modeling part. In Section V, we examine a test case that has several S&D requirements: Secure Service Discovery. In Section VI, we review most related works that address pattern development. Finally, Section VII concludes this paper with a short discussion about future works.

II. Foundations and Conceptual Framework

The following subsection presents briefly the SEMCO framework, describes an example in order to illustrates the issues identified in the paper. Then, the structure of the pattern modeling is presented.

A. SEMCO Approach

SEMCO is a a federated modeling framework and the goal of Fig. 1 is to highlight the notion of integrated repository of metamodels to deal with system engineering. The proposed approach is to use an integrated repository of models to capture several concerns of safety critical embedded systems namely extra and non functional properties.

These artifacts will be used to capture in order to model all the facets of the system and its parts: logical (software and hardware components) and the infrastructure. They are provided as informal textual document, as semi-formal document using UML, SysML and Eclipse modeling framework, and as formal document using formal frameworks.

Currently, as shown in Fig. 1, SEMCO defines and provides 13 different artifacts types representing different engineering concerns and architectural information.

SEMCO approach promotes the use of patterns as first-class artifacts to embed solutions of extra-functional concerns such as safety, security and performance requirements of systems, specify the set of correct configurations, and capture the execution infrastructure of the systems, supporting the mechanisms to implement these concerns.

In this paper, we focus on the study of *pattern* artifact to deal with extra-functional concerns as recurring design problems in specific design contexts, and to present a well-proven generic scheme for their solutions. For that, we propose a language for modeling design patterns to get a common representation of patterns for several domains in the context of safety critical systems applications. Therefore, such a solution allows to capture appropriate characteristics of design concerns and to utilize several views. We begin describing our motivating example.

B. Motivating Example: Authorization Pattern

The essence of Fig. 4 is to promote the separation of general-purpose services from implementations. In our context, this figure highlights the separation of general-purpose of the pattern from its required mechanisms. This is an important issue to understand the use of patterns to target extra-functional concerns. In which layer related mechanisms are placed depends on the assurance a client has in how the services are in some particular layer. As example of a common and a widely used patterns, we choose the *Authorization Pattern* [13].

For instance, in a distributed environment in which users or processes make requests for data or resources, this pattern describes who is authorized to access specific resources in a system, in an environment in which we have resources whose access needs to be controlled. As depicted in Fig. 2, it indicates how to describe allowable types of accesses (authorizations) by active computational entities (subjects) to passive resources (protection objects). Such a pattern provides support to define possible ways of uses that applies to every level of the system.

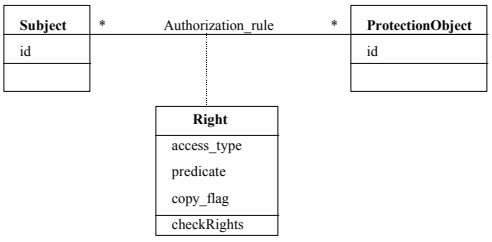


Fig. 2. Authorization Pattern

However, those authorization patterns are slightly different with regard to the application domain. For instance, a system domain has its own mechanisms and means to serve the implementation of this pattern using a set of protocols ranging

 ISBN: 978-1-61208-158-8

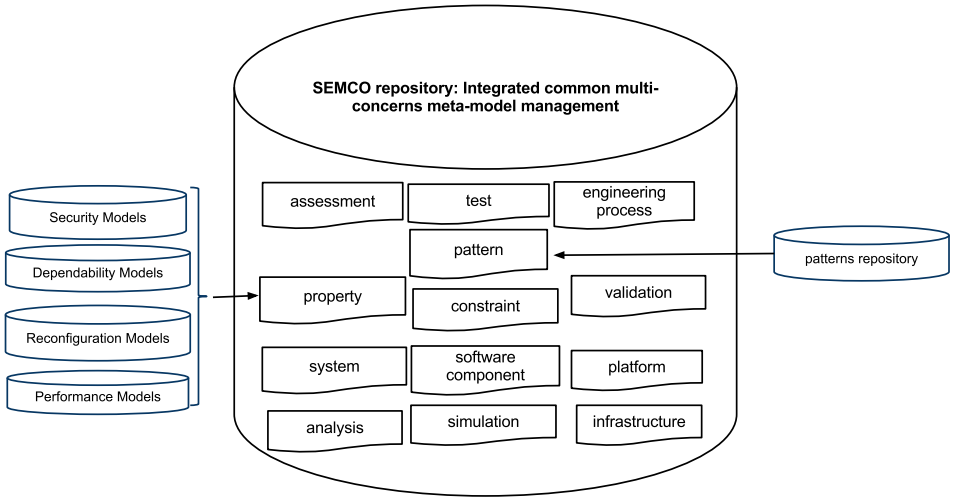


Fig. 1. SEMCO Generic Structure

from RBAC (Role Based Acess Control), FireWall, ACL'S (Acess Control List), Capabilities, and so on. For more breadth and depth, the reader is referred to [15]. In summary, they are similar in the goal, but different in the implementation issues for instance to determine for each active entity that can access resources, which resources it can access, and how it can access them. So, the motivation is to handle the modeling of patterns by following abstraction. In the followings, we propose to use *Capabilities* [15] to specialize the implementation of the authorization pattern. This solution is already used at the hardware and operating system level to control resources.

More specifically, the access rights of subjects with respect to objects are stored in an *access control matrix* M. each subject is represented by a row and each object is represented by a column. An entry in such a matrix $M[s, o]$ contains precisely the list of operations subject s are allowed to request on object o. More efficient way to store the matrix is to distribute the matrix row-wise by giving each subject a list of capabilities it has for each object. Without such a capability for a specific object means that the subject has no access rights for that object. Then, requests for resources are intercepted and validated with the information in the capabilities. The interception and the validation are achieved by a special program usually referred to as *reference monitor*. For instance, whenever a subject s requests for the resource r of object o, it sends such a request passing its capability. The reference monitor will check whether it knows the subject s and if that subject is allowed to have the requested operation r, as depicted in Fig. 3. Otherwise the request fails. It remains the problem of how to protect a capability against modification by its holder. One way is to protect such a capability (or a list of them) with a signature handed out by special certification authorities named *attribute certification authorities*.

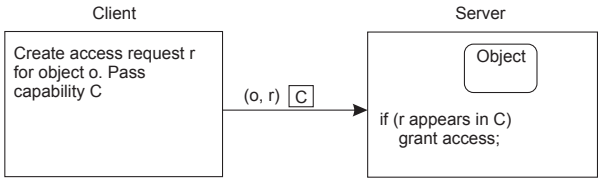


Fig. 3. Protecting Resources using Capabilities

C. Pattern Metamodel Structure

One of the major considerations in designing multi-concerns safety critical systems is to determine at which level of abstraction concerns should be placed. The supporting research includes specification, modeling, implementation mechanisms, verification, etc. to name a few. For example, distributed systems are organized into separate layers following some reference models, e.g., applications, middleware and the operating system services. Combining the layered organization of target applications, domain specific systems and patterns modeling leads roughly to what is shown in Fig. 4.

The framework must cope with multi-concerns and domain specific properties. For this purpose, the proposition presented in this paper is based on a MDE approach and on three levels of abstraction: (i) Pattern Fundamental Structure (PFS), (ii) Domain Independent Pattern Model (DIPM) and (iii) Domain Specific Pattern Model (DSPM). Firstly this decomposition aims at allowing the design of multi-concerns applications in the context of safety (since combining extra-functional concerns and domain specific artifacts introduces a great complexity), and secondly to overcome the lack of formalism of the classical pattern form (e.g., textual).

III. Pattern Modeling Language

This section is dedicated to present our pattern modeling framework. As we shall see, the originality of this approach

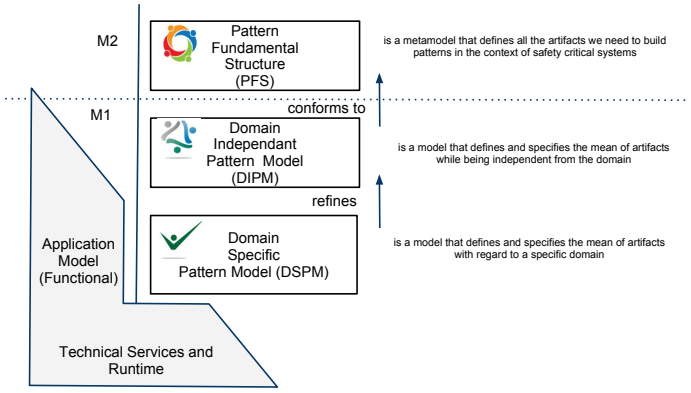


Fig. 4. Pattern Modeling Framework Structure

is to consider patterns as building blocks that expose services (via interfaces) to deal with concerns (properties). This pattern definition provides a clear and flexible structure. Moreover, the modularity it enables allows to tame the complexity of large systems. As introduced into the last section, our pattern modeling language is based on three levels of abstraction. We start the description with a template inspiring our proposal (the patterns modeling language). Then, the first level of abstraction, namely PFS, will be described.

A. Patterns Documenting Model: Template

For our best knowledge, there is no consensus about the required information to represent patterns in the domain of software engineering, particularly when dealing with extra-functional properties. For this reason, we propose the following template. Note, however, that our proposition is based on GoF, and we deeply refined it in order to fit with the non-functional needs.

B. Pattern Fundamental Structure

The Pattern Fundamental Structure (PFS), as depicted in Fig. 6, is a meta-model which defines a new formalism for describing patterns and which constitutes the base of our pattern modeling language. Such a formalism describes all the artifacts (and their relations) required to capture all the facets of patterns. Here we consider patterns as building blocks that expose services (via interfaces) and manage properties (via features) yielding a way to capture meta-information related to patterns and their context of use. These pattern are specified by means of a domain-independent generic representation and a domain-specific representation. The next section details the principal classes of our meta-model, as described with UML notations in Fig. 6. Fig. 7 depicts in more details the meaning of principal concepts used to edit a pattern.

IV. Modeling Patterns

As mentioned earlier, our modeling framework promotes to use three levels of abstraction: (i) Pattern Fundamental Structure (PFS), (ii) Domain Independent Pattern Model (DIPM) and (iii) Domain Specific Pattern Model (DSPM). In this section, the required artifacts will be pointed out while following the two abstraction levels (i.e., DIPM and DSPM). These two levels with a authorization pattern presented in Section II-B are illustrated. Note, however, that for lack of space we only specify those principle elements.

- *Name*. Meaningful word or phrase for the pattern to facilitate the documentation. This unique name gives a first idea about the pattern purpose,
- *Also known as*. Describes known occurrences of the pattern,
- *Related Patterns*. Here related patterns are mentioned
- *Example*. To show the use of the pattern,
- *Context*. To describe the conditions where the problem occur,
- *Problem*. Informal description of a problem that needs an appropriate solution,
- *Solution*. The solution describes how to solve the problem,
- *Precondition*. The set of conditions have to be met in order to be able to use the pattern,
- *Postcondition*. The impact of the pattern integration,
- *Attributes*. The set of information to configure and customize the pattern,
- *Properties*. The kind of properties the pattern provides to resolve the requirements.
- *Constraints*. This part describes constraints for a reasonable and correct use of the pattern.
- *Structure*. Indicates with class, sequence, and other UML diagrams, the form of the solution.
- *Interfaces*. To encapsulate patterns interaction functions. The way the pattern interact with its environment.

Fig. 5. Extra-Functional Pattern Template

A. Domain Independent Pattern Model (DIPM)

This level is intended to generically represent patterns independently from the application domain. This is an instance of the PFS. As we shall see, we introduce new concepts through instantiation of existing concepts of the PFS meta-model in order to cover most existing patterns in safety critical applications. In our case study, the DIPM of the authorization pattern consists of two communicating entities. The authorization pattern is defined as followed:

- *Properties.* At this level, we identify one property: *confidentiality.*
- *External Interfaces.* The authorization pattern exposes its functionalities through function calls:
  - $request(S, AT, PR)$: the subject S sends request about access type AT concerning the protected resource PR.
- *Internal Structure.* The behavioral of authorization pattern can be modeled by a UML Sequence Diagram following Fig. 2.

B. Domain Specific Pattern Model (DSPM)

The objective of the specific design level is to specify the patterns for a specific application domain. This level offers artifacts at down level of abstraction with more precise *information* and *constraints* about the target domain. This modeling level is a refinement of the DIPM, where the specific

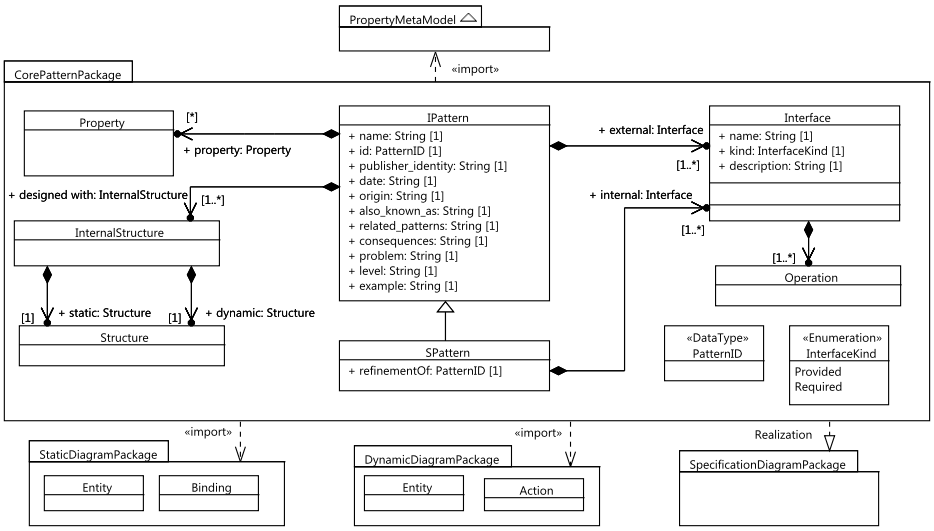


Fig. 6. Pattern Fundamental Structure

characteristics and dependencies of the application domain are considered. Different DSPM would refine a same DIPM for all needed domain. For instance, when using Capabilities as a mechanism related to the application domain to refine the authorization pattern at DSPM, we introduce the following artifacts:

- *Properties.* In addition to the refinement of the property identified in the DIPM, at this level, we consider: *deny unauthorized access*, *permit authorized access*, and *efficiency* properties.
- *External Interfaces.* This is a refinement of the DIPM external interface:
  $request(S, AT, PR, C)$: the subject S sends request about access type AT concerning the protected resource PR passing its capability C.

- *Internal Interfaces.* Let the subset of functions related to the use of capabilities to refine the authorization pattern:
  - $sign(C)$: the certification authority signs the capability C,
  - $verifyCert()$: the attribute capability certificate is verified,
  - $extractCap()$: the capability is extracted from the certificate,
  - $checkRight(S, AT, PR, C)$: the reference monitor verifies, using the capability, whether PR appears in the C.
- *Internal Structure.* The behavioral of authorization based on capability and reference monitor can be modeled by a UML Sequence Diagram following the description in Section II-B.

V. Secure Service Discovery for Home Control

In the following, an example will illustrate the approach point defined in the previous sections. The current trend aims at integrating more intelligence into the homes to increase services to the person. For this purpose, electronics equipments are widely used while providing easy and powerful services. However, to integrate all the services automatically requires a plug and play like system. For this issue, this example aims at providing a pattern for home control domains which provides a secure service discovery. Compared to a usual service discovery, this pattern will use a secure channel in order to protect all data. Fig. 8 shows two use cases: (i) adding a new equipment (ii) updating the current configuration. The Fig. 8 is described in form of UML notations and highlights the interfaces of the pattern in order to support the two main use cases. In the next subsections, the interface and the static internal structure will be pointed out while following the two abstraction levels (i.e., DIPM and DSPM) proposed by the paper.

A. Representation at DIPM: Person using a remote InternetBox.

Regarding to the interface, it is necessary to declare some operations which allow the user to check if new implementations exist (i.e., update) and to detect the context (i.e., new equipment). Then, it is necessary to define all properties addressed by the pattern.

Fig. 9 illustrates the representation of secure connection at DIPM. At this level, we deal with a person using a remote InternetBox. As mentioned in the previous section, we choose *Authorization pattern* (or Access control) for security property. Regarding the internal structure of the pattern, we consider the following: a person uses a multi-media device which communicates with an internetBox via a Wifi connection.

B. Specialize a pattern through the DSPM: Subscriber using a remote OperatorBox

The interfaces must be adapted in order to match with the specific communication used in the domain. Regarding to the properties, at the DIPM, we only specify a very generic security property. At this level, it is possible to refine this property by defining the mechanisms, the length of the

- *IPattern*. this block represents a modular part of a system that encapsulates a solution of a recurrent problem. An *IPattern* defines its behavior in terms of provided and required interfaces. As such, an *IPattern* serves as a type whose conformance is defined by these provided and required *interfaces*. Larger pieces of a system's functionality may be assembled by reusing patterns as parts in an encompassing pattern or assembly of patterns, and wiring together their required and provided interfaces. An *IPattern* may be manifest by one or more artifacts, and in turn, that artifact may be deployed to its execution environment. The *IPattern* has some fields to describe the related particular recurring design problem that arises in specific design contexts. These fields are based on the GoF [5] information as described in Fig 5. This is the key entry artifact to model pattern at domain *independent* level.
- *Interface*. *IPattern* interacts with its environment with *Interfaces* which are composed of *Operations*. An *IPattern* owns provided and required interfaces. A provided interface is implemented by the *IPattern* and highlights the services exposed to the environment. A required interface corresponds to services needed by the pattern to work properly. So, larger pieces of a system's functionality may be assembled by reusing patterns as parts in an encompassing pattern or assembly of patterns, and wiring together required and provided interfaces. Finally, we consider two kinds of interface:
  - *External interfaces* allow implementing interaction with regard to the integration of a pattern into an application model or to compose patterns.
  - *Internal interfaces* allow implementing interaction with the platform. For instance, at a low level, it is possible to define links with software or hardware module for the cryptographic key management. These interfaces are realized by the *SPattern*. Please,note an *IPattern* does not have *InternalInterface*.
- *Property*. is a particular characteristic of a pattern related to the concern dealing with.
- *Internal Structure*. constitutes the implementation of the solution proposed by the pattern. Thus the *InternalStructure* can be considered as a white box which exposes the details of the *IPatterns*. In order to capture all the key elements of the solution, the *Internal Structure* is composed of two kinds of *Structure*: *static* and *dynamic*. Please, note that a same pattern would be have several possible implementations[a].
- *SPattern*. inherits from *IPattern*. It is used to build a pattern at DSPM. Furthermore a *SPattern* has *Internal Interfaces* in order to interact with the platform. This is the key entry artifact to model pattern at domain *specific* level.

[a]Usually referred to as variants of design patterns.

Fig. 7. Pattern MetaModel Dependencies

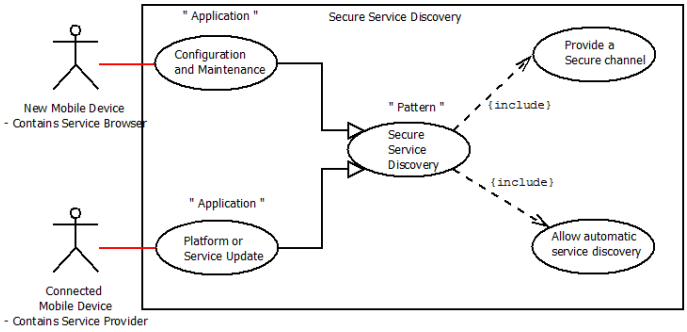


Fig. 8. Secure Service Discovery Use Cases

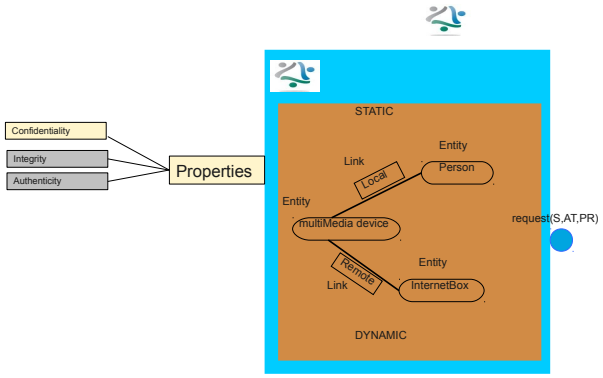


Fig. 9. Service Discovery example: Person using a remote InternetBox (DIPM)

keys, etc. Moreover, it is possible to add new properties. For instance, a RCES property can be added like the cryptographic time.

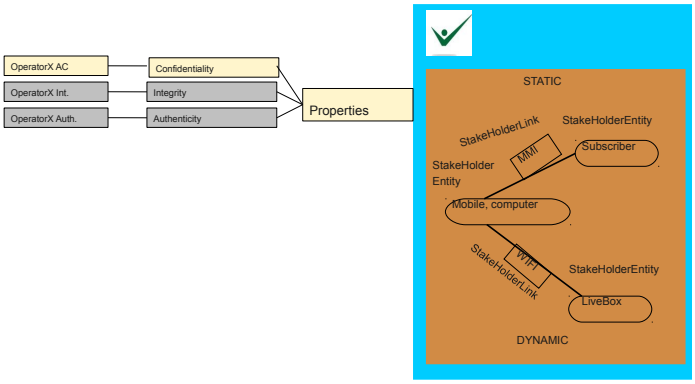


Fig. 10. Person using a remote InternetBox (DSPM)

Fig. 10 illustrates the representation of secure connection at DSPM. At this level, the pattern deals with an operator subscriber using a 'Operato'Box. For instance, we choose OperatorX AC for *Access Control*. The information expressed

by the internal structure of the pattern is the following: an operator subscriber person uses a computer which is connected to a 'Operator'Box.

VI. Related Works

Design patterns are a solution model to generic design problems, applicable in specific contexts. Since their appearance, and mainly through the work of Gamma et al [5], they have attracted much interest. The supporting research includes domain patterns, pattern languages and their application in practice. Several tentatives exist in the literature to deal with patterns for specific concern [17], [7], [18], [3], [16]. They allow to solve very general problems that appear frequently as sub-tasks in the design of systems with security and dependability requirements. These elementary tasks include secure communication, fault tolerance, etc. The pattern specification consists of a service-based architectural design and deployment restrictions in form of UML deployment diagrams for the different architectural services.

To give a flavor of the improvement achievable by using specific languages, we look at the pattern formalization problem. *UMLAUT* [8] is an approach that aims to formally model design patterns by proposing extensions to the UML meta model 1.3. They used OCL language to describe constraints (structural and behavioral) in the form of meta collaboration diagrams. In the same way, *RBML(Role-Based Meta modeling Language)* [10] is able to capture various design perspectives of patterns such as static structure, interactions, and state-based behavior. The framework *LePUS* [6] offers a formal and visual language for specifying design patterns. It defines a pattern in an accurate and complete form of formula with a graphical representation. A diagram in LePUS is a graph whose nodes correspond to variables and whose arcs are labeled with binary relations.

With regard to the integration of patterns in software systems, the *DPML (Design Pattern Modeling Language)* [11] allows the incorporation of patterns in UML class models. Recently, [14] explains how pattern integration can be achieved by using a library of precisely described and formally verified security and dependability solutions. Other domain specific solutions as [7], [18] exist.

While many patterns for specific concern have been designed, still few works propose general techniques for patterns. For the first kind of approaches [5], design patterns are usually represented by diagrams with notations such as UML object, annotated with textual descriptions and examples of code. There are some well-proven approaches [4] based on Gamma et al. However, this kind of techniques does not allow to reach the high degree of pattern structure flexibility which is required to reach our target. The framework promoted by LePUS [6] is interesting but the degree of expressiveness proposed to design a pattern is too restrictive.

To summarize, in software engineering, design patterns are considered as effective tools for the reuse of specific knowledge. However, a gap between the development of the system and the pattern information still exists. This becomes more exciting when dealing with specific concerns namely security and dependability for several application sectors.

VII. Conclusion and Future Work

Extra-functional and non-functional concerns are not building blocks added to an application at the end of the life cycle. It is necessary to take into account this concern from the requirement to the integration phase. In this paper, we promote the use of patterns to provide practical solutions to meet these requirements and follow a MDE-based approach to specify such patterns. Indeed, MDE solutions allows to meet several concerns around one model while ensuring coherence between all businesses.

Here, we propose a common pattern modeling language to design multi-concerns safety critical system applications. This kind of application requires an adapted language to design it. Indeed, a classical form of pattern is not sufficient to tame the complexity of such application – complexity occurs because of both the concerns and the domain management. To reach this objective and to tame this complexity, our language is based on an advanced form of pattern using a MDE approach. The proposed approach is structured in 3-layer architecture. The first one corresponds to a metamodel which defines a generic structure of patterns. Then, two other layers are an instance of the metamodel. The two last levels allows us to integrate domain specific features at the end of the process.

The benefit of this structure is to offer a common language for different domain application. So far, this common language encompasses four industrial sectors, namely, home control, industry control, automotive, and metering [9].

As a side remark, note that our goal is to obtain an even high level abstraction to represent patterns to capture several facets of extra-functional and non-functional concerns in the different domain of safety critical systems applications, not an implementation of a specific solution. The key is then to show that the major sectors of such systems applications dealing with such concerns become covered by our approach. This result raises new and previously unanswered questions about general techniques to model these kind of patterns. We believe that this result is of particular interest to build a multi-concerns systems discipline that is suited to a number of sectors in safety critical systems.

The next step of this work consists in implementing other patterns including those for security, safety, reconfiguration and dependability to build a repository of multi-concerns patterns. Another objective for the near future is to provide guidelines concerning both the integration of all the presented results in a more global process with the pattern life cycle (i.e., create, update, store patterns) and the integration of pattern in an application. All patterns are stored in a repository. Thanks to it, it is possible to find a pattern regarding to concern criteria. At last, guidelines will be provided during the pattern development and the application development (i.e., help to choose the good pattern).

 ISBN: 978-1-61208-158-8

Acknowledgments. This work is initiated in the context of SEMCO framework. It is supported by the European FP7 TERESA project and by the French FUI 7 SIRSEC project.

REFERENCES

[1] OMG Unified Modeling LanguageTM (OMG UML), superstructure version2.2, 2009. Version 2.2 is a minor revision to the UML 2.1.2 specification. It supersedes formal/2007-11-02.

[2] G. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. *Pattern-Oriented Software Architecture: a system of patterns*, volume 1. John Wiley and Sons, 1996.

[3] F. Daniels. The reliable hybrid pattern: A generalized software fault tolerant design pattern. In *PLOP 97*, 1997.

[4] B. P. Douglass. *Real-time UML: Developing Efficient Objects for Embedded Systems*. Addison-Wesley, 1998.

[5] E. Gamma, R. Helm, R. E. Johnson, and J.Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[6] E. Gasparis, J. Nicholson, and A. H. Eden. Lepus3: An object-oriented design description language. In *In: Gem Stapleton et al. (eds.) DIAGRAMS, LNAI 5223*, pages 364–367, 2008.

[7] V. Di Giacomo and al. Using security and dependability patterns for reaction processes. pages 315–319. IEEE Computer Society, 2008.

[8] A. L. Guennec, G. Sunyé, and J-M. Jézéquel. Precise modeling of design patterns. In *In Proceedings of the third International Conference on the Unified Modeling Language (UML'2000)*, pages 482–496. Springer-Verlag, 2000.

[9] B. Hamid, N. Desnos, C. Grepet, and C. Jouvray. Model-based security and dependability patterns in rces: the teresa approach. In *Proceedings of the International Workshop on Security and Dependability for Resource Constrained Embedded Systems,SD4RCES '10*, pages 1–4.

[10] D k Kim, R. France, S. Ghosh, and E. Song. A uml-based metamodeling language to specify design patterns. In *Patterns, Proc. Workshop Software Model Eng. (WiSME) with Unified Modeling Language Conf. 2004*, pages 1–9, 2004.

[11] D. Mapelsden, J. Hosking, and J. Grundy. Design pattern modelling and instantiation using dpml. In *CRPIT '02: Proceedings of the Fortieth International Conference on Tools Pacific*, pages 3–11. Australian Computer Society, Inc., 2002.

[12] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady. Security in embedded systems: Design challenges. *ACM Trans. Embed. Comput. Syst.*, 3(3):461–491, 2004.

[13] M. Schumacher, E. Fernandez, D. Hybertson, and F. Buschmann. *Security Patterns: Integrating Security and Systems Engineering*. John Wiley & Sons, 2005.

[14] D. Serrano, A. Mana, and A-D Sotirious. Towards precise and certified security patterns. In *Proceedings of 2nd International Workshop on Secure systems methodologies using patterns (Spattern 2008)*, pages 287–291. IEEE Computer Society, September 2008.

[15] A. S. Tanenbaum and M. Steen. *Distributed systems, principles and paradigms, 2/E*. Prentice-hall, Inc, 2007.

[16] M. Tichy and al. Design of self-managing dependable systems with uml and fault tolerance patterns. In *WOSS '04: Proceedings of the 1st ACM SIGSOFT workshop on Self-managed systems*, pages 105–109. ACM, 2004.

[17] J. Yoder and J. Barcalow. Architectural patterns for enabling application security. In *Conference on Pattern Languages of Programs (PLoP 1997)*, pages 1–31, 1998.

[18] N. Yoshioka, H. Washizaki, and K. Maruyama. A survey of security patterns. *Progress in Informatics*, pages 35–47, 2008.

 ISBN: 978-1-61208-158-8

A Formalization of UML AD Refinement Patterns in Event B

Ahlem Ben Younes
Research Unit of Technologies of Information and Communication (UTIC)- ESSTT
Tunisia
Ahlem.benyounes@fst.rnu.tn

Leila Jemni Ben Ayed
Research Unit of Technologies of Information and Communication (UTIC)- ESSTT
Tunisia
Leila.jemni@fsegt.rnu.tn

***Abstract*— In this paper, we propose a specification and verification technique using the combination UML Activity Diagrams (AD) and Event B to both improve graphical representation of the workflow application structure and its functions. Its required properties are also verified. The workflow is initially expressed incrementally graphically with UML AD, then translated into Event B and verified using the B powerful support tools. The Event-B expression of the UML AD model allows us to give it a precise semantics. We propose a workflow applications constructive approach in witch Event B models are built incrementally from UML AD models, driven by UML AD refinement patterns. The use of the B formal method and its refinement mechanism allows the verification of the correction of the UML AD refinement patterns.**

Keywords-Progressive Development; Workflow Applications; Specification; Refinement UML AD; Patterns; Event-B; Formal Verification.

I. INTRODUCTION

The Workflow Applications (WA) are characterized by a high complexity. Increasingly, they became omnipresent in the critical calculation domain (natural or industrial disasters) and they have to obey to the realiabity and safety requirements. Thus, the need of an adequate software specification technique and a suitable development method is increased. The used specification formalisms need to be comprehensive, expressive, and precise.

A workflow is a set of activities (tasks/ process) that are ordered according to a set of procedural rules to achieve a result or a goal. A workflow model (workflow specification) is the definition of a workflow. A workflow is either an atomic task, known as elementary task/activity or a sub-workflow (nesting), a composite activity/task.

Traditional workflow models have obvious shortcomings in describing complex workflows. Such complexity is due not only to the hierarchical property of business process, but also to the complicated dependencies among tasks. Composition is an important approach to model larger and more complex workflow application. Task refinement is one kind of workflow composition approaches.

Indeed, specifying a complex system is a difficult task, which cannot be done in one step. The stepwise refinement technique facilitates the understanding of complex systems by dealing with the major issues before getting involved in the details. It consists of developing the system through different levels of abstraction. The system under development is first described by a specification at a very high level of abstraction. A series of iterative refinements may then be performed with the aim of producing a specification, consistent with the initial one, in which the behavior is fully specified and all appropriate design decisions have been made. Stepwise software development can be fully exploited only if the language used to create the specifications is equipped with formal refinement machinery, making it possible to prove that a given specification S_C is a refinement of another specification S_A.

The Unified Modeling Language Activity Diagrams (UML AD) [3] are considered as an Object Management Group (OMG) [15] standard notation in the area of workflow applications modelling. The idea of one standard language for modelling provides many advantages to software development, such as simplified training and unified communication between development teams.

In our work, an UML activity diagrams approach based on stepwise refinement technique for the workflow specification is proposed. The refined workflow is presented in UML using the hierarchical capabilities of the UML Activity Diagram notation [4][5]. Workflow's hierarchy comes from the hierarchy of processes goals (Task). Goals or activities of workflow applications are organized in a hierarchy obtained from the Sequence/And/Or refinement of higher level activities (goals) into lower-level activities (atomic task). To describe the decomposition of the activity, we propose some patterns that allow to model some refinement of activity: sequence, choice (OR), parallel (AND), loop. Thus, the description of the workflow at different levels of abstraction becomes possible. In addition, our objective is to provide a specification and verification technique for workflow applications using UML AD, which give readable models and an appropriate formal method allowing verification of required properties (such no deadlock) to prove the correctness of the workflow specification.

Indeed, the main problem with UML activity diagrams is that they have no formal semantics and in consequence UML AD does not allow the formal verification of functional workflow applications properties (safety, deadlock-inexistence, liveness, fairness, etc) and the correction of the patterns.

On the other hand, the Event B method [2] is a variant of the B formal method [1], proposed by Abrial to deal with

 ISBN: 978-1-61208-158-8

distributed, parallel and reactive systems [2]. The concept of refinement is the key notation for developing B models. The refinement of a formal model allows one to enrich the model in step by step approach. The last refinement gives the implementation machine, which map directly to a programming language such as C or ADA. B models provide an automatic proof, which convinces the user that the system is effectively correct and satisfies properties, which are presented as invariants/assertions. The strong point of B is support tools like as AtelierB [6] or B4free [7], an academic version of AtelierB. Most theoretical aspects of the method, such as the formulation of Proof Obligations (PO), are carried out automatically. The automatic and interactive provers are also designed to help designer to discharge the generated proof obligations. All of these points make B well adapted to large scale industrial projects [8]. However, B is still difficult to learn and to use. In addition, there is a lack of methodological studies related to the incremental development of complex system using the refinement mechanisms.

This is why a graphical representation of B models is required. For that purpose, we propose a constructive approach in witch Event B models are built incrementally from UML AD models, driven by UML refinement patterns.

Our work presents a specification and verification technique using the combination UML AD and Event B to both improve graphical representation of the workflow application structure and its functions such as the complex properties, and also verify required properties.

In our approach, the workflow is initially expressed incrementally graphically with UML AD refinement patterns, then translated into Event B and verified using the B powerful support tools.

The Event-B expression of the UML AD model allows us to give it a precise semantics. In this context, there have been efforts for defining semantics for activity diagram in the works of Eshuis [10][11]. However, these works not consider the hierarchical decomposition of activities in UML AD. In addition, from the validation point of view, in our approach, the verification of WA is based on a proof technique and therefore it does not suffer from the state space explosion occurring in classical model checking as in the cases of works in [9] [10] [11] and [12].

Our contribution, in this context, consists of using Event B method and its associate refinement process to encode the hierarchical decomposition of activities in UML AD: Each decomposition of a complex activity in UML AD is translated into Event B by refining the event corresponding to this activity. This refinement introduces the decomposition defined in the original UML AD workflow specification. Thus, a step by step UML AD workflow description and validation is performed in parallel.

Refinement allows the developer to express the relevant properties at the refinement level where they are expressible. Then, further refinements will preserve these properties avoiding reproving them again.

The use of the B formal method and its refinement mechanism allows the verification of the correction of the UML AD refinement patterns by the use of the B support tools.

This paper continues our previous works [4][5] by addressing the Event-B formalization of UML AD patterns (sequence, parallel, choice) for workflow applications with additional studies, results and proofs. In [4][5], we have only proposed translations rules for UML AD notation into Event B models. In these innovative works, we propose a formal framework to define refinement patterns for UML AD. We define an Event-B semantic for each UML AD refinement pattern for WA by constructing set-theoretic mathematical models (See Section 4 and 5). Based on the classical set of inference rules from Event-B [13], we identify the systematic proof obligations for each UML AD activity refinement pattern. The use of the B formal method and its refinement mechanism allows the verification of the correction of the patterns. The Event-B formalization of the other UML AD models is a work in progress.

The remainder of the paper is organized as follows: Section 2 presents a brief overview of the semi-formal UML activity diagrams notation. Section 3 presents a brief overview the formal Event B method. Section 4 details our proposed approach that consists in expressing a UML AD model with Event-B. Section 5 illustrates the approach by presenting the Event-B formalization of the sequence refinement pattern. Finally, a summary of our work concludes the paper.

II. UML ACTIVITY DIAGRAMS

An activity diagram is a variation of a state machine in which the states represent the execution of actions or subactivities and the transitions are triggered by the completion of the actions or subactivities. We use activity diagrams to model computational, communication, and synchronization operations/process of parallel and distributed applications. Moreover, we use the hierarchical decomposition (thanks to refinement) offered by UML activity diagrams to model complexes applications gradually in incremental way on several levels (see Figure 2). An action state is used to model a step in the execution of an algorithm (atomic action), or a workflow process (Subactivity represents a composed activity). A subactivity state invokes an activity diagram. When a subactivity state is entered, the activity diagram nested in it, which corresponds to the refined activity, is executed. A subactivity state is shown in the same way as an action state with the addition of an icon in the upper left corner depicting a nested activity diagram (see Figure 1.(b)). Transitions are used to specify that the flow of control (the token) pass from one action to the next. An activity diagram expresses a decision when guard conditions are used to indicate different possible transitions (see Figure 1.(a)). A guard condition specifies a condition that must be satisfied in order to enable the firing of an associated transition. A merge has two or more incoming transitions and one outgoing transition. It can be used to merge decision branches back together. Fork and join are used to model parallel flows of control (see Figure 1.(b)). The initial and final state are, respectively, visualized as a

 ISBN: 978-1-61208-158-8

solid ball and a solid ball inside a circle. Figure 1.(a) illustrates how to model a loop by employing an activity diagram, whereas Figure 1.(b) shows one option for modeling the parallel execution of two activities.

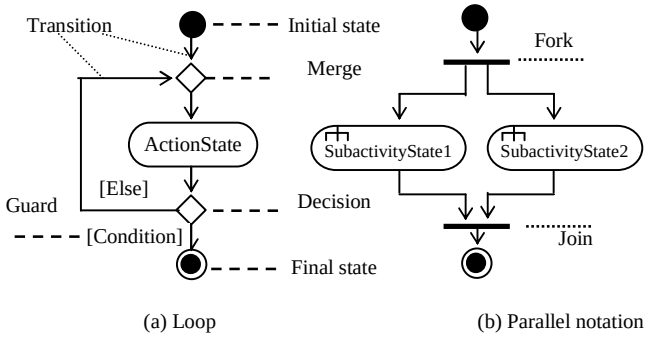


Figure 1. UML activity diagram notation

One of the main features of UML is the refinement with hierarchical decomposition of activities in UML AD, which permits to obtain a detailed specification from an initial specification. Figure 2 illustrates how to model complex application like distributed and parallel application, on several levels, by employing a refinement technique of UML AD.

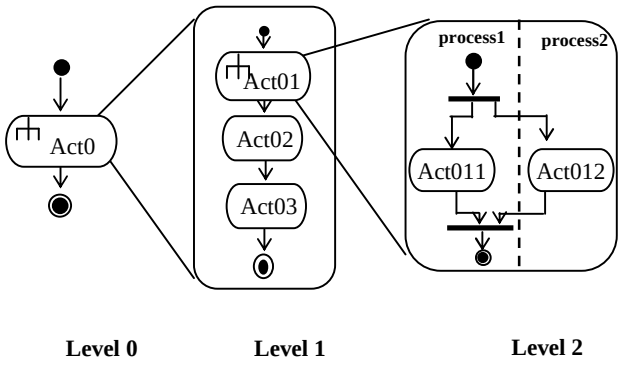


Figure 2. Hierarchical decomposition of activities in UML activity diagrams

Our choice of UML AD is motivated by the fact that workflow modelling is strongly supported by UML through activity diagrams [1]. Moreover, UML is easy to read and understand by human beings.

In the Section 4, we show how the semantics of activity diagrams can be formally described in Event B.

III. EVENT B METHOD

We use the B method [1] and its event-based definition [2] to formalize UML AD models of workflow application.

Event B model: Development in the Event B method is based on the concept of model [2]. A model shown below is composed of:

MODEL < *name*>
VARIABLES
<*variables*>
INVARIANT
< *invariant*>
ASSERTIONS
< *assertion*>
INITIALISATION
<*initialization of variables*>
EVENTS
< *events*>
END

- Descriptive specification, which describes what the system does using a set of variables, constants, properties over constants and invariants which specify required properties to be verified in each state. This constitutes the static definition of the model.

- Operational specification, which describes the way how the system operates, it is composed of an initial state and various transitions (events) which show how the set of variables of the descriptive specification can move in time.

An Event B model is composed of set atomic events described by particular generalized substitution (**ANY**, **BEGIN** and **SELECT**). Each event Evt is fired if the guard P associated to this event is true. For the purpose of this paper, we will only use the **SELECT** substitution Evt= **SELECT** P **THEN** G **END**. Moreover, a B model contains a set of properties i.e invariants, liveness, safety and reachability properties which can be prove during the development thanks to the embedded proof system associated to B and the tool supported by B4free [6].

Refinement of Event B models: Each Event B model can be refined. A refined model is defined by adding new events, new variables and a gluing invariant. Each event of the abstract model is refined in the concrete model by adding new information describing how the new set of variables and the new events evolve. All the new events appearing in the refinement refine the skip event of the refined model.

 ISBN: 978-1-61208-158-8

The gluing invariant ensures that the properties expressed and proved at the abstract level (in the **ASSERTIONS** and **INVARIANTS** clauses) are preserved in the concrete level. Moreover, **INVARIANT, ASSERTIONS** and **VARIANT** [14] clauses express deadlock and livelock.

1. They shall express that the new events of the concrete model are not fired infinitely (no livelock). A decreasing variant is introduced for this purpose.
2. They shall express that at any time, an event can be fired (no deadlock). This property is ensured by asserting (in the **ASSERTIONS** clause) that the disjunction of all the abstract events guards implies the disjunction of all the concrete events guards.

At every step of the refinement, proof obligations ensure that events and initialization preserve the system invariant. A set of proof obligations that is sufficient for the correctness must be discharged when a refinement is postulated between two B components [2] [14].

A strong point of the B method is that the B support tools like B4free [7] provides utilities to discharge automatically the generated proof obligations (of the invariant preservation and the refinement correctness). Analyzing the non-discharged proof obligations with the B support tools is an efficient and practical way to detect errors encountered during the specification development.

Moreover, in the refinement, it is not needed to reprove these properties again while the model complexity increases. Notice that this advantage is important if we compare this approach to classical model checking where the transition system describing the model is refined and enriched.

Finally, the choice of Event-B is due to its similarity and complementarity with UML AD: both Event-B and UML AD have the notion of refinement (constructive approach).

IV. THE PROPOSED APPROACH

A. *Presentation*

Our approach relies on the following steps:

- Step 1: Initially, the workflow is modeled graphically with UML AD refinement patterns.
- Step 2: For current decomposition level, the resulting graphical readable model is translated into Event B applying the approach described in [4][5].
- Step 3: This Event B model is enriched by relevant properties (no deadlock, no livelock, etc) which are defined in the INVARIANTS and ASSERTIONS clauses. These properties will be proved using the B4free tool [7].
- Step 4: We isolate the events of the Event B model whose POs, associated to the introduced invariant of step 3, are not provable.
- Step 5: The UML AD model of step1 is re-design by introducing a UML AD scope embedding the events identified at step 4 and a compensation/fault handler component.
- Step 6: Apply step 2.

This step-based approach is applied until the associated Event B model is free of unproved PO.

B. *Formalization of UML AD*

To achieve our objective, we formalize with Event-B the UML AD refinement patterns that analysts use to generate a UML AD workflow hierarchy.

The UML AD language [3] offers two categories of activities:

1) Atomic activities (action) representing the primitive operations performed by the process. They are defined by action node in UML AD.

2) Composed activities representing the sub-workflows (nesting), obtained by composing primitive activities and/or other composed activities using the sequence, parallel (For/Join), choice (Decision/ Merge) control constructs in UML AD.

In the remaining of this paper, we refer to the decomposition (refinement) of a composed activity by the activities it contains as a result of the refinement operation using refinement patterns.

In this innovative work, the formal assertion defining an activity A is written in first-order logic. Thus, the general form of the assertions associated to the activities is A-Pre => A-post where A-Pre and A-Post are predicates associated to an activity A (See Figure3). Symbol => denotes the classical logical implication. Such assertions state that from a state in which A-Pre holds, we must reach another state in which A-Post holds.

If we refer to the concepts of guard and postcondition that exists in Event-B, a UML AD activity can be considered as a postcondition of the system, since it means that a property must be established. Following our pervious works [4], we have proposed to express each UML AD activity as a B event, where the action represents the achievement of the activity. Then, we will use the Event-B refinement relation and additional custombuilt proof obligations to derive all the subactivity of the system by mean of B events.

At the high level of abstraction, there is only one event for representing the parent activity. In accordance with the Event-B semantics, if the guard of the event is true, then the event necessarily occurs. For the new events built by refinement and associated to the subactivity, we guarantee by construction that no events prevent the postconditions to be established. For that, we have proposed an Event-B semantic for each UML AD refinement pattern by constructing set-theoretic mathematical models. Based on the classical set of inference rules from Event-B [13], we have identified the systematic proof obligations for each UML AD activity refinement pattern.

 ISBN: 978-1-61208-158-8

To better illustrate the approach, the next section presents just the Event-B refinement semantics related to the sequence refinement pattern.

V. THE FORMALIZATION OF THE UML AD SEQUENCE REFINEMENT PATTERN

The sequence activity refinement pattern refines a composed activity by introducing intermediate sequence states A01,..., A0n for reaching a state satisfying the target condition (denoted by A0-Post) from a state satisfying the current condition (denoted by A0-Pre) as shown in Figure 3 (with just two sub-activity).

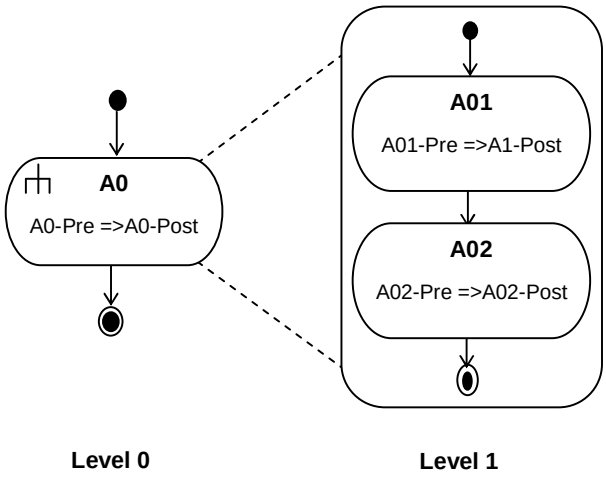


Figure 3. Sequence activity refinement pattern

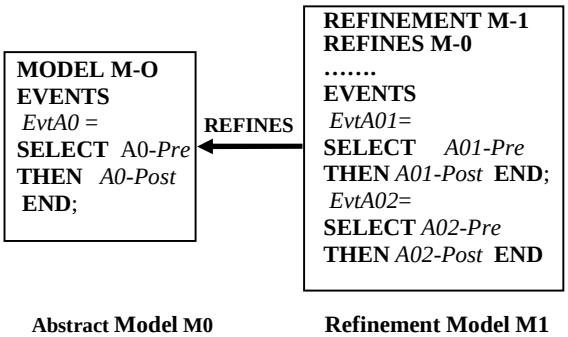


Figure 4. Overview of the Event-B representation of the UML AD model

A. Description

The first sub-activity A0 is an activity with the sequence condition as target condition; it states that the sequence condition (denoted by A0-Pos) must hold if.

The specific current condition A01-Pre (which can be larger than the current condition A0-Pre of the parent activity) holds in the current state. The second sub-activity states that the specific target condition A02-Post (which can be larger than the target condition A0-Post of the parent goal) must hold if the specific sequence condition A02-Pre (derived from A01-Post) holds in the current state.

B. Formal definition

As explained in the last section, each level i ($i \in [0..n]$) is represented in the hierarchy of the UML AD models as an Event-B model Mi that refines the model $Mi-1$ related to the level $i - 1$. Moreover, we represent each activity A j,i ($j \in$ [0,..,n] activity index) as a B event EvtAi,j , where the guard is the transcription of A-Guard from the activity expression, and the **THEN** part is the translation into Event-B of A-Post (see Figure 4).

C. Proof obligations identification

We are going to give systematic rules defining exactly what we have to prove for this pattern in order to ensure that each concrete event (EvtA01, EvtA02) indeed refines its abstraction EvtA0. In fact, we have to prove three different lemmas:

- The ordering constraint (PO1) expresses the sequence characteristic between the Event-B events. PO1 ensures that the target condition of the activity A01 implies the current condition of the activity A02.

$$\text{A01-Post} \Rightarrow \text{A02-Pre} \quad \textbf{(PO1)}$$

- The guard strengthening (PO2) ensures that the concrete guard is stronger than the abstract one. In other words, it is not possible to have the concrete version enabled whereas the abstract one would not. The term “stronger” means that the concrete guard implies the abstract guard.

$$\text{A01-Pre} \Rightarrow \text{A0-Pre} \quad \textbf{(PO2)}$$

- The correct refinement (PO3) ensures that the sequence of concrete events transforms the concrete variables in a way which does not contradict the abstract event.

$$\text{A02-Post} \Rightarrow \text{A0-Post} \quad \textbf{(PO3)}$$

 ISBN: 978-1-61208-158-8

The Event-B refinement semantics of the sequence refinement pattern requires to prove three proof obligations ((PO1) (PO2) (PO3)) that could easily be discharged by the current version of B4free or Rodin automatic theorem prover.

VI. CONCLUSION

We have proposed a specification and verification technique for workflow applications using UML AD and Event B. The workflow is at first modelled with UML AD refinement patterns, which is understandable allowing communications with costumers, then translated the resulting model into Event B, which is enriched by relevant properties (Safety, nodeadlock) to be verified using powerful support tool B4free[7]. This approach allows to rigorously verify UML specifications by analysing derived B specifications and to prove that the modelled workflow using the AD respects all safety and reliability constraints by the formal verification of its properties. Analyzing derived B specifications (thanks to B4free tool) is a practical and rigorous way to improve initial UML AD specifications.

Our contribution consists in the use of the Event B method and its associate refinement process to encode the hierarchical decomposition of activities in UML AD and its tools for the formal verification of workflow applications. In addition, the strong point in our approach is that the validation can be performed at any development stage and particularly at early steps allowing saving at development. Thus, a step by step UML AD workflow description and validation is performed in parallel.

For an incremental development of AW using UML AD, we have proposed some activity refinement patterns (sequence, parallel, choice). In this paper, we have proposed a formal framework to define refinement patterns for UML AD. The use of the B formal method and its refinement mechanism allows the verification of the correction of the patterns by the B support tools.

In contrast to the works of Eshuis [10] [11], Karamanolis [12] and Van der Aalst [9], in our works, the verification is based on a proof technique and therefore it does not suffer from the state number explosion occurring in classical model checking as in the cases of their works.

Actually, we are actively working on the extension of our works to investigate new refinement patters presented in [4] and to generalize our method. In addition, in future work, we envisage the formal validation of our transformation rules.

REFERENCES

[1] J.-R, Abrial, “The B Book. Assigning Programs to Meanings”. Cambridge University Press, 1996.

[2] J.-R.Abrial. ”Extending B without changing it” (for developing distributed systems)”. In H Habrias, editor, First B Conference. 1996.

[3] R. Johason, I. Jacobson, and G. Booch, “The Unified Modelling Language reference Manual” .Addison- Wesley. 1998.

[4] A. Ben Younes, and L.-Jemni,Ben Ayed “ Using UML Activity Diagrams and Event B for Distributed and Parallel Applications”. In 31st Annual IEEE International Computer Software and Applications Conference .COMPSAC 2007: pp, 163-170.

[5] A. Ben Younes, and L.-Jemni, Ben Ayed, “Specification and verification of Workflow Applications using Combination of UML Activity Diagrams and Event B’’. In The 5th International Conference on Software Engineering and Data Technologies. ICSOFT 2010: pp 312-316.

[6] Clearsy, “ System Engineering Atelier B”. Version 3.6. ,2001

[7] Clearsy“B4free,” Available at www.b4free.com . (June 30, 2011).

[8] P. Behm, P. Desforges, and J.-M. Meynadier. “MÉTÉOR: An Industrial Success in Formal Development”. April 1998. An invited talk at the 2nd Int. B conference, LNCS 1939.

[9] W.M.P. van der Aalst. “Workflow Verification: Finding Control-Flow Errors using Petri-net-based Techniques”. In Business Process Management: Models, Techniques, and Empirical Studies, volume 1806 of Lecture Notes in Computer Science, pp 161-183. Springer-Verlag, Berlin, 2000.

[10] R. Eshui and R. Wieringa. “Tool Support for verifying UML Activity Diagram”. IEEE transaction on software Engineering , volume 30 , N°7; pp 437-447. 2004.

[11] R. Eshuis and R. Wieringa. “ A formal semantics for UML activity diagrams”. Technical Report TR-CTIT-01-04, Centre for Telematics and Information Technology, University of Twente, 2001.

[12] C. Karamanolis, D.Giannakopoulou, J. Magee, and S. M.Wheater “Formal verification of workflow schemas”. University of Newcastle, Technical Report. 2000.

[13] J.-R. Abrial. Chapter 2 of the forthcoming book: "Modeling in Event-B: System and Software Engineering Forthcoming book". http://www.event-b.org/A ch2.pdf.

[14] C. Metayer, J,-R. Abrial, and L. Voisin. ”Event B language’’, Technical Report D7, RODIN Project Delivrable. 2005.

[15] Object Management Group (OMG), http://www.omg.org. (June 30, 2011).

 ISBN: 978-1-61208-158-8

A Quality Control System using Texture Analysis in Metallurgy

Jonathan M. Blackledge
School of Electrical Engineering Systems
Dublin Institute of Technology
Dublin, Ireland
Email: jonathan.blackledge@dit.ie

Dmitriy A. Dubovitskiy
Radiation and Environmental Science Centre
Dublin Institute of Technology
Dublin, Ireland
dmitriy.dubovitskiy@dit.ie

***Abstract*—Object detection, recognition and texture classification is an important aspect of many industrial quality control systems. In this paper, we report on a system designed for the inspection of surfaces which has a range of applications in the area of metallurgy. The approach considered is based on the application of Fractal Geometry and Fuzzy Logic for texture classification and, in this paper, focuses on the manufacture of rolled steel. The manufacture of high quality metals requires automatic surface inspection for the assessment of quality control. Quality control systems are required for several tasks such as screening defected products, monitoring the manufactures process, sorting information for different applications and product certification and grading for end customers. The system discussed in this paper was developed for the Novolipetck Iron and Still Corporation in Russia and tested with images captured at a rolling mill with metal sheets moving at speed of up to six meters per second and inspected for several defect classes. The classification method used is based on the application of a set of features which include fractal parameters such as the Lacunarity and Fractal Dimension thereby incorporating the characterisation of surface surfaces in terms of their texture. The principal issues associated with texture recognition are presented which includes fast segmentation algorithms. The self-learning procedure for designing a decision making engine using fuzzy logic and membership function theory is also presented and a new technique for the creation and extraction of information from a membership function considered. The methods discussed, and the system developed, have a range of applications in 'machine vision' and automatic inspection. However, in this publication, we focus on the development and implementation of a surface inspection system that can be used in a iron and steel manufacture by non-experts to the automatic recognition system operators.**

Keywords-Computer vision; patterns analysis; segmentation; object recognition; self-learning; fuzzy logic; image morphology.

I. INTRODUCTION

Pattern recognition is a part of image analysis, which involves the use of image processing methods that are often designed in an attempt to provide a machine interpretation of an image, ideally, in a form that allows some decision criterion to be applied [1], [2]. Pattern recognition uses a range of different approaches that are not necessarily based on any one particular theme or unified theoretical approach. This is because there is no complete and unique theoretical model available for explaining and simulating the processes of visual image comprehension by humans. Hence, machine vision remains a rather elusive subject area in which automatic inspection systems are advanced without having a fully operational theoretical framework as a guide. Nevertheless, numerous algorithms for understanding two- and three-dimensional objects in a digital image have and continue to be researched in order to design systems that can provide reliable automatic object detection, recognition and classification in an independent environment, [9], [10], [11] and [13].

Machine Vision can be thought of as the process of linking parts of the visual object's field with stored information or 'templates' with regard to a pre-determined significance for the observer. There are a number of questions concerning vision such as: (i) what are the goals and constraints? (ii) what type of algorithm or set of algorithms is required to effect vision? (iii) what are the implications for the process, given the types of hardware that might be available? (iv) what are the levels of representation required to achieve vision? The levels of representation are dependent on what type of segmentation and edge detection can and/or should be applied to an image. For example, we may be able to produce primal sketches from an image via some measure of the intensity changes in a scene. These are recorded as place tokens and stored in a database. Regions of pixels with similar intensity values or sets of lines are obtained by isolating the edges of an image scene and computed by locating regions where there is a significant difference in the intensity. Such sets are subject to inherent ambiguities when computed from a given input image and associated with those from which an existing data base has been constructed. These ambiguities can only be overcome by the application of high-level rules, based on how humans interpret images, but the nature of this interpretation is not defied. Parts of an image will tend to have an association if they share size, colour, figural similarity, continuity, shading and texture. For this purpose, one needs to consider how best to segment an image and what form this segmentation should take.

The identification of the edges of features on metal surfaces (as given in Figure 1, for example) is an important component for developing quality control system in metallurgy. This identification provides information on the basic topology of a feature from which an interpretative match can

 ISBN: 978-1-61208-158-8

be achieved. Some edges can be detected only in terms of a representative view a whole image and have no connection with local pixels. Nevertheless, the segmentation of an image into a complex of edges is a useful pre-requisite for object identification and the solution may require analysis of the whole scene. Although many low-level processing methods can be applied for this purpose, the problem is to decide which object boundary each pixel in an image falls within and which high-level constraints are necessary. In many cases, a principal question is, which comes first, recognition or segmentation?

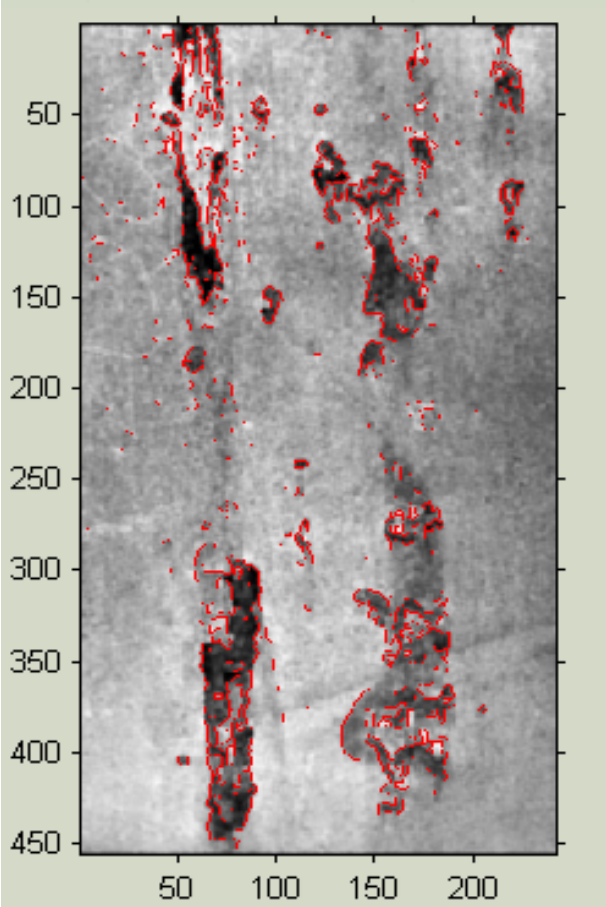


Figure 1. Example of a metal surface with edge-based features.

Compared to image processing, computer vision (which incorporates machine vision) is more than automated image processing. It results in a conclusion, based on a machine performing an inspection of its own. The machine must be programmed to be sensitive to the same aspects of the visual field as humans find meaningful. Segmentation is concerned with the process of dividing an image into meaningful regions or segments. It is used in image analysis to separate features or regions of a pre-determined type from the background; it is the first step in automatic image analysis and pattern recognition. Segmentation is broadly based on one of two properties in an image: (i) similarity; (ii) discontinuity. The first property is used to segment an image into regions which have grey or colour levels within a predetermined range. The second property segments the image into regions of discontinuity where there is a more or less abrupt change in the values of the grey or colour levels.

In this paper, we consider an approach to object detection in an image scene that is based on a new segmentation edge recognition or edge tracing or edge following algorithm. The segmented object is then analysed in terms of metrics derived from both a Euclidean and Fractal geometric perspective, the output fields being used to train a fuzzy inference engine with a supervised leaning technique, the recognition structure being based on some of the technologies for image processing, analysis and machine vision reported in [12]. The approach considered is generic in that it can, in principle, be applied to any type of imaging modality. The system developed includes features that are based on the textural properties of an image which is an important theme is patterns analysis.

II. Pattern Recognition

Pattern recognition can be considered to be a form of machine understanding based on assigning a particular class to an object. The tasks of construction and application of formal operations for numerical or character representation of objects of a real or ideal world is the basis of pattern recognition. This depends on establishing equivalence relations that express a fit of evaluated objects to any class with independent semantic units. The recognition classes of equivalence can be set by the user in the construction of an algorithm, which uses own pithy representations or external padding information on a likeness and difference of objects in the context of a solved task; the basis for phrase 'recognition with the teacher'. For a typical object recognition system, the determination of the class is only one of the aspects of the overall task. In general, pattern recognition systems receive data in the form of 'raw' measurements which collectively form a stimuli of 'feature' vector [3], [4]. Uncovering relevant attributes in the elements present within the feature vector is an essential part of such systems. An ordered collection of such relevant attributes which more clearly represent the underlying features of the object is assembled into the feature vector. In this context, learning amounts to the determination of rules of associations between the features and attributes of a pattern.

Practical image recognition systems generally contain several stages in addition to the recognition engine itself. The recognition represents information processing that is realised by some converter of the information (by an intellectual information channel), having an input and output. On input, such a system establishes information on the properties of an object. On output, the information shows which class or feature of an object is assigned. When a computerised

 ISBN: 978-1-61208-158-8

system decides on the task of classification without engaging external learning information, it is called automatic classification - 'recognition without the teacher'. The majority of algorithms for pattern recognition require the engagement of a number of considerable computational capabilities, which can be provided only with high-performance computer equipment [5].

There are two principal methods for object recognition solutions with a parametric and non-parametric approach. Statistical voting and alphabetic propositions has been reviewed in [7][8][12]. The main disadvantage with this approach is that classes have to be clearly defined so that no overlapping is allowed. Methods based on a principal of separation and potential functions can be found in [6] and [11]. A large amount of training data or preliminary information about system is required which makes the recognition process less flexible. In general, there is no system which considers objects from the point of view of a superposition of global scenery. This leads to the following problem: how can we evaluate an object in terms of it being part of the 'bigger picture' without losing specific details on its particular texture for precise recognition? This paper attempts to solve this problem by merging concepts from Fractal Geometry [21] [22] [23] [24] [25] and Fuzzy Logic [15] [16] [17]. We start by considering the problem of object location.

III. Object Location

Recognition is the process of comparing individual features against some pre-established template subject to a set of conditions and tolerances. This task can be reduced to the construction of some function determining a degree of proximity of the object to a sample - a 'template' of the object. The process of recognition commonly takes place in four definable stages: (i) image acquisition and filtering; (ii) object location (with edge detection); (iii) measurement of object parameters; (iv) object class estimation and decision making.

Suppose we have an image which is given by a function $f(x, y)$ and contains some object described by a set of features $S = \{s_1, s_2, ..., s_n\}$. We consider the case when it is necessary to define a sample, which is somewhat 'close' to this object in terms of a matching set. The system discussed in this paper is based on an object detection technique that includes a novel segmentation method and must be adjusted and 'fine tuned' for each area of application. This includes those features associated with an object for which fractal models are well suited [1], [2], [21]. A conventional method consists of calculating some function of a pointwise coincidence between the map of the object and the image together with a search for the maximum of this function. In terms of a 'similarity function', this method can be represented in terms of metrics that include the sum of square deviations, the sum of the modulus of deviations or as a pair of sum of multiplications of values of brightness (function of the greatest transparency), for example. The first two similarity functions compute the 'smallness' of a functional pair; instead of searching for a maximum it is necessary to search for a minimum.

Not all fragments of an object are equally important for recognition and hence, a broadly distributed functional evaluation matched with weighted coefficients can be undertaken on separate parts. Appropriate similarity functions can be used as a sum of the weighted squares of deviations, a sum of the weighted modules of deviations and the sum of the weighted multiplication of pairs of brightness values. The correct selection of weight coefficients is important in the field of identification and can be calculated from a given set of samples. The common application for weighted comparisons occurs in the field of artificial neural networks. The advantage of usage of neural networks lies in the capability of introducing a flexible set of weights during operation (system training). This property becomes especially important if a set is based on a non-stationary model which varies in time while it is extended and updated.

The system described in this paper provides a decision using a knowledge database by subscribing different objects. The 'expert data' in the application field creates a knowledge database by using a supervised training system with a number of model objects [15]. At this stage, the learning technique uses positive feedback for the second step of object location and filtering. We consider an image of a metal surface as given Figure 1.

Figure 1 represents the result after applying a conventional filtering and edge detection procedure. The conventional method does not provide continuous edges in order to locate a feature. We have therefore designed a new object location procedure that considers the image in its entirety without detailing smaller features. This is based on a measure of weight coefficients to provide information about object connectivity. The result of this procedure can be given in Figure 2.

The calculation of weight coefficients for each pixel is defined as $k_{x,y}$:

$$f_{m,n} = f(x, y)k_{x,y}$$

where

$$k_{x,y} = \left(\frac{1}{f(x,y)} \begin{bmatrix} k_{x-1,y+1} & k_{x,y+1} & k_{x+1,y+1} \\ k_{x-1,y} & p_{x,y} & k_{x+1,y} \\ k_{x-1,y-1} & k_{x,y-1} & k_{x+1,y-1} \end{bmatrix} \right) * p_{\mathrm{obj}(x,y)}$$

There is local dependency between the current pixel $f_{m,n}$ and the object pixels. The global evaluation is determined by $p_{\mathrm{obj}(x,y)}$ which is the probability that the pixel could be a part of an object. This probability is calculated from a fuzzy logic membership function which has a loop-back to the current object location. The function $p_{\mathrm{obj}(x,y)}$ is a two

 ISBN: 978-1-61208-158-8

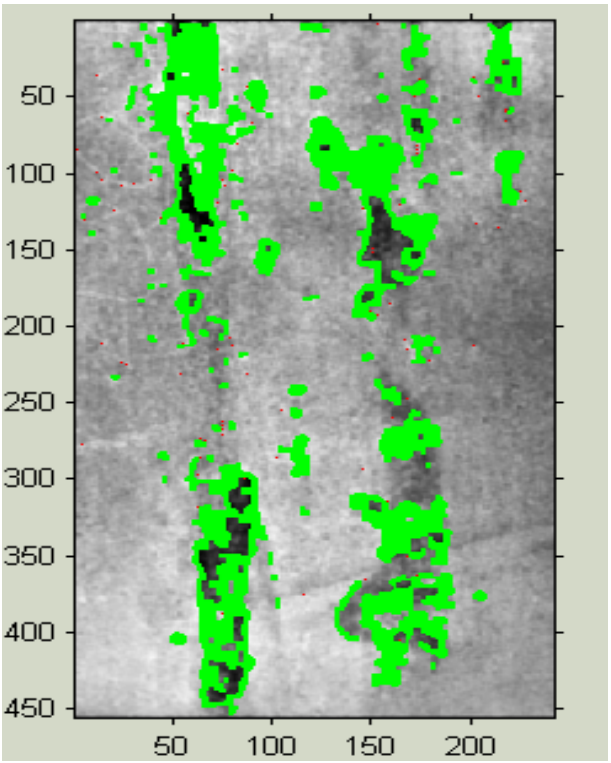


Figure 2. Metal surface with approximate object locations.

dimensional matrix and recalculates local values dynamically using the object table location $f_{m,n}$. The construction of this matrix is based on the following: The intensity level of the objects is computed. This level uses only those pixels which have not been recognised as a part of an object. To start with, the object level denoted by $\mathrm{L_{obj}}$ is higher than the background level $\mathrm{L_{bgr}}$ as the recognition process continues. As long as $\mathrm{L_{obj}} == \mathrm{L_{bgr}}$, all objects are recognised as having been indexed according to the equation [19], [20]

$$\mathrm{L_{bgr}} = \mathrm{mean}[f(x,y) - f(m,n)]$$

In order to obtain $\mathrm{L_{obj}}$, a probabilistic min-max equation, which has been experimentally tested for different surfaces, is used given by [19]:

$$L_{\mathrm{obj}} = \begin{cases} L_x, & L_x \geq L_y; \\ L_y, & \text{otherwise.} \end{cases}$$

where

$$L_x = \frac{1}{2}\left(\min_y\left(\max_x f(x,y)\right) - \langle \max_x f(x,y)\rangle_y\right) + \langle \max_x f(x,y)\rangle_y,$$

$$L_y = \frac{1}{2}\left(\min_x\left(\max_y f(x,y)\right) - \langle \max_y f(x,y)\rangle_x\right) + \langle \max_y f(x,y)\rangle_x.$$

In order to maintain simplicity, we do not include in this equation that component which is responsible for dividing previously defined objects in $f_{m,n}$. For more complex images, the user can define a region of interest *a priori*.

The second stage is to compute a particular value of membership function $p_{\mathrm{obj}(x,y)}$ according to the equation

$$p_{\mathrm{obj}(x,y)} = \int_{xy} (f_{x,y}\mathrm{L_{obj}} - \mathrm{L_{bgr}} + \mathrm{edge}_{xy})dxdy$$

for the closed border of the object. The function edge_{xy} is an edge detection function. Depending on the application, special filters including the 'Detour by object contour' and 'Convex Hull Spider' can be included [20]. Information from the application of these filters can be stored and used for the classification and decision making procedure. Each object is enumerated in terms of the procedural steps associated with the object recognition process.

IV. Decision Making Processes

Information about feature classes is stored in a Knowledge Data Base (KDB) which is composed of probability coefficients for a particular class. The class probability is a vector $\mathbf{p} = \{p_j\}$ which is estimated from the object feature vector $\mathbf{x} = \{x_i\}$ and membership functions $m_j(\mathbf{x})$ defined in the knowledge database. If $m_j(\mathbf{x})$ is a membership function, then the probability for each j^{th} class and i^{th} feature is given by

$$p_j(\mathbf{x}_i) = \max\left[\frac{\sigma_j}{|\mathbf{x}_i - \mathbf{x}_{j,i}|} \cdot m_j(\mathbf{x}_{j,i})\right]$$

where σ_j is the distribution density of values $\mathbf{x}_j$ at the point $\mathbf{x}_i$ of the membership function. The next step is to compute the mean class probability given by

$$\langle p\rangle = \frac{1}{j}\sum_j \mathbf{w}_j p_j$$

where $\mathbf{w}_j$ is the weight coefficient matrix. This value is used to select the class associated with

$$p(j) = \min\left[(p_j \cdot \mathbf{w}_j - \langle p\rangle) \geq 0\right]$$

providing a result for a decision associated with the j^{th} class. The weight coefficient matrix is adjusted during the learning stage of the algorithm.

The decision criterion method considered represents a weighting-density minimax expression. The estimation of the decision accuracy is obtained by using the density function

$$d_i = |\mathbf{x}_{\sigma_{\max}} - \mathbf{x}_i|^3 + [\sigma_{\max}(\mathbf{x}_{\sigma_{\max}}) - p_j(\mathbf{x}_i)]^3$$

with an accuracy determined by

$$P = \mathbf{w}_j p_j - \mathbf{w}_j p_j \frac{2}{\pi}\sum_{i=1}^{N} d_i.$$

 ISBN: 978-1-61208-158-8

The overall accuracy depends on the level of confidence of an expert. In some cases, an expert is unable to make clear decisions about which class belongs to an object. In such cases, it must be 50/50 in order for the system to consider this case as overlapping, but not to delete it and use extra data from the KDB to make a decision.

Consider a sample belonging to one of three groups: Scale, Cleavage crack or Cusping. We then undertake the same operations as those during the training session. The system then finds the object, computes its fractal dimension which, in this case study, is 2.58, for example, and the convexity factor (0.69). The degree of confidence determined by all the parameters functions is displayed in Figure 3.

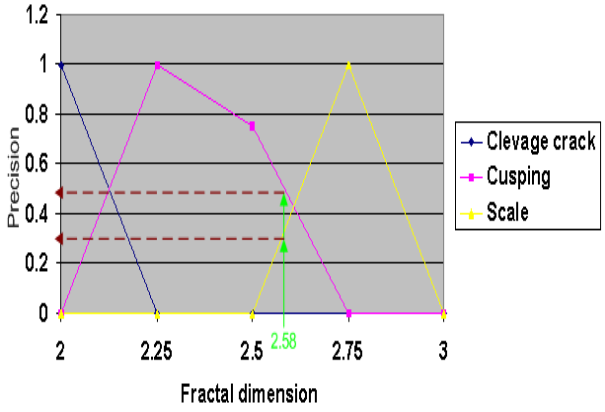


Figure 3. Precision definitions

We compute the degree of confidence for each class as:

(Scale)=0.27+0.87=1.14

(Cusping)=0.46+0.42=0.88

(Clevage crack)=0+0=0

The maximum of these values characterizes that class to which the given image corresponds. In the example given, the output is *Scale*. For industrial systems with many reference classes, it is possible to utilise scaling factors for each of the computed parameters in conformity with a measure of influence (weight coefficient) on a parameter for a class definition. The weight coefficients will be automatically readjusted with the next teaching input. Once the expert decides to correct some class performance, then the corresponding input parameters will be reconsidered for chosen class only.

The computation time depends on the image resolution, normally varying from 2 to 10 seconds in the MatLab environment. For a metal surface moving at 6m/sec, the algorithm described above would need to be implemented by means of a field-programmable gate array (FPGA), which will lead to the computation fitting within frames.

V. Conclusion and Future Works

This paper has been concerned with the task of developing a methodology and applications that are concerned with two

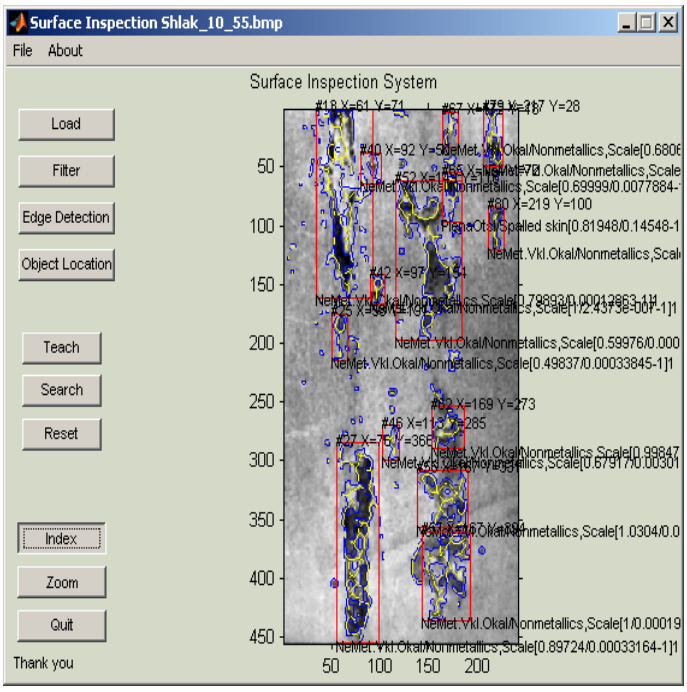


Figure 4. Result of surface inspection.

key tasks: (i) the partial analysis of an image in terms of its fractal structure and the fractal properties that characterize that structure; (ii) the use of a fuzzy logic engine to classify an object based on both its Euclidean and fractal geometric properties. The combination of these two aspects has been used to define a processing and image analysis engine that is unique in its modus operandi but entirely generic in terms of the applications to which it can be applied.

The work reported in this paper is part of a wider investigation into the numerous applications of pattern recognition using fractal geometry as a central processing kernel. This has led to the design of a new library of pattern recognition algorithms including the computation of parameters in addition to those that have been reported here such as the information dimension, correlation dimension and multi-fractals [21]. The inclusion or otherwise of such parameters in terms of improving vision systems such as the one considered here remains to be understood. However, from the work undertaken to date, it is clear that texture based analysis alone is not sufficient in order to design a recognition and classification system. Both Euclidean and fractal parameters need to be combined into a feature vector in order to develop an operational vision system, which includes objects that have textural properties such as those associated with medical imaging.

The creation of logic and general purpose hardware for artificial intelligence is a basic theme for any future development based on the results reported in this paper for the applications developed and beyond. The results of the

 ISBN: 978-1-61208-158-8

current system can be utilized in a number of different areas although medical imaging would appear to be one of the most natural fields of interest because of the nature of the images available, their complex structures and the difficulty of obtaining accurate diagnostic results which are efficient and time effective. A further extension of our approach is to consider the effect of replacing the fuzzy logic engine used to date with an appropriate Artificial Neural Network. It is not clear as to whether the application of an ANN could provide a more effective system and whether it could provide greater flexibility with regard to the type of images used and the classifications that may be required.

ACKNOWLEDGMENT

The authors are grateful for the advice and help of Dr P.P. Chernov (Novolipetsk Iron and Steel Corporation), Professor V. Deviatkov and Professor A. Chernikov (Bauman Moscow State Technical University).

REFERENCES

[1] J. M. Blackledge, *Digital Signal Processing*, 2nd Edition, Horwood Publishing, 2006.

[2] J. M. Blackledge, *Digital Image Processing*, Horwood Publishing, 2005.

[3] W. E. L. Grimson, *Object Recognition by Computers: The Role of Geometric Constraints*, MIT Press, 1990.

[4] B. D. Ripley, *Pattern Recognition and Neural Networks*, Academic Press, Oxford, 1996.

[5] R. B. Macy Abhijit and S.Pandya, *Pattern Recognition with Neural Networks in C++*, IEEE Press, Florida Atlantic University, 1995.

[6] C. T. Leondes, *Image Processing and Pattern Recognition*, Academic press, London, 1998.

[7] E. L. Hall, *Computer Image Processing and Recognition*, Academic press, London, 1979.

[8] T. Pavlidis, *Algorithms for Graphics and Image Processing*, Computer Science Press, USA, 1982.

[9] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*, Academic press, London, 1997.

[10] H. Freeman, *Machine Vision. Algorithms, Architectures, and Systems*, Academic press, London, 1988.

[11] J. Louis and J. Galbiati, *Machine Vision and Digital Image Processing Fundamentals*, State University of New York, New-York, 1990.

[12] R. Boyle, M. Sonka and V. Hlavac, *Image Processing, Analysis and Machine Vision*, PWS, USA, 1999.

[13] W. E. Snyder and H. Qi, *Machine Vision*, Cambridge University Press, England, 2004.

[14] V. S. Nalwa and T. O. Binford, *On Detecting Edges* IEEE Trans. Pattern Analysis and Machine Intelligence, (PAMI-8), 699-714, 1986.

[15] L. A. Zadeh, *Fuzzy Sets and their Applications to Cognitive and Decision Processes*, Academic Press, New York, 1975.

[16] E. H. Mamdani, *Advances in Linguistic Synthesis of Fuzzy Controllers*, J. Man. Mach., 8, 669-678, 1976.

[17] E. Sanchez, *Resolution of Composite Fuzzy Relation Equations*, Int. Control, 30, 38-48, 1976.

[18] N. Vadiee, *Fuzzy Rule Based Expert Systems-I*, Prentice Hall, Englewood, 1993.

[19] J. M. Blackledge and D. Dubovitsky, *A Surface Inspection Machine Vision System that Includes Fractal Analysis*, ISAST Transactions on Electronics and Signal Processing Vol. 3, No 2, 76-89, 2008.

[20] J. M. Blackledge and D. Dubovitsky, *Object Detection and Classification with Applications to Skin Cancer Screening*, ISAST Transactions on Intelligent Systems, Vol. 1, No 1, 34-45, 2008

[21] M. J. Turner, J. M. Blackledge and P. R. Andrews, *Fractal Geometry in Digital Imaging*, Academic Press, London, 1998.

[22] B. B. Mandelbrot, *The Fractal Geometry of Nature*, W.H.Freeman, New York, 1983.

[23] K. Falconer, *Fractal Geometry*, Wiley, 1990.

[24] N. Sarkar, B. Chaudhuri and P. Kundu, *Improved Fractal Geometry Based Texture Segmentation Technique*, IEEE Proceedings Vol.140 , number 5, pages 233-241, 1993.

[25] J. Keller and S. Chen, *Texture Description and Segmentation through Fractal Geometry*, Computer Vision Graphics and Image Processing, number 45, pages 150-166, 1989.

[26] Cancer research uk, accessed 12.09.2011 http://cancerhelp.cancerresearchuk.org/type/melanoma/

[27] J. S. Lim, *Two-Dimensional Signal and Image Processing*, Prentice-Hall, 1990.

[28] J. M. Blackledge and D. Dubovitsky, *Morphological Analysis from Images of Hyphal Growth using a Fractional Dynamic Model*, Theory and Practice of Computer Graphics, University of Warwick, 2011, p17-24, 2011.

[29] J. M. Blackledge and D. Dubovitsky, *Moletest: A Web-based Skin Cancer Screening System*, The Third International Conference on Resource Intensive Applications and Services, INTENSIVE 2011, May 22-27, Venice, Italy, 22-29, 2011.

[30] J. M. Blackledge and D. Dubovitsky, *Pattern Recognition in Cytopathology for Papanicolaou Screening*, Theory and Practice of Computer Graphics, Sheffield 6-8 September, Vol. 25, No. 1, 2010.

 ISBN: 978-1-61208-158-8

Prediction of Distortion Patterns in Image Steganography by Means of Fractal Computing

Shanyu Tang
Faculty of Computing
London Metropolitan University
London, UK
s.tang@londonmet.ac.uk

YongFeng Huang
Department of Electronic Engineering
Tsinghua University
Beijing, China
yfhuang@tsinghua.edu.cn

***Abstract*—This paper describes a new method of predicting image distortion patterns in image steganography by using fractal computing. The method uses the successive random addition algorithm to simulate the image distortion patterns of embedding a secret image in an 'innocent' cover image, testing the fractal-like behaviour of image distortion patterns. The distortion patterns identified in the experimental data are characterised by a fractal Hurst parameter, which can be used to make predictions of future trends.**

Keywords-distortion patterns; image steganography; fractal computing

I. INTRODUCTION

Over the last three decades, people have sought ways to protect sensitive information against attack to make sure it is not received by unintended recipients. Conventional security measures are built on encryption, which encodes data such that an unintended recipient cannot determine its intended meaning. Encryption is now confronted with serious challenges since the increase in computational power has led to decryption of several classic encryption algorithms, indicating vulnerabilities in the encryption primitives. A major drawback to encryption is that the existence of data is not hidden. A solution to this problem is digital steganography [1].

Digital steganography is a form of data hiding in which a secret message is hidden within another file. It is essentially about embedding a message or file in another 'cover' file, called the carrier file, such that the carrier file is not altered enough to raise suspicion that something may be hidden within it [2]. Data to be hidden is the carrier medium; the cover file in which the data is hidden is the steganographic medium. Both parties communicating via steganography must use the same stego application.

Steganography in 'static' cover objects, such as plaintext, image files with BMP or JPEG format, and audio files in WAV or MP3 format, has been explored extensively [3] [4]. See [5] for a good survey of such techniques. There have been efforts to develop the steganalysis techniques for detecting steganography in static cover objects such as text, image or audio files. A larger number of image and audio steganalysis methods have been reported in the literature [6]-[9].

Previous image steganography studies have been largely focused on developing algorithms for steganography in image files and the steganalysis techniques, with no or little attention being given to image distortion patterns. In fact, image distortion is an intuitive indication of imperceptibility of image steganography. An understanding of image distortion patterns could help find ways to improving data embedding capacity, i.e. the number of bits in a byte of the cover image that can be replaced without affecting the image, which is a bottleneck to image steganography.

The use of data mining techniques and neural networks tools enables us to analyse digital information and understand future patterns, which may occur [10] [11]. As an example of applications, data mining techniques detect particular patterns in customer behaviour and future trends [12]. In this study, we attempt to use fractal computing methods to study whether image distortion patterns in steganography have burstiness behaviour - bursting on many or all scales. The burstiness behaviour is analogous to the self-similar or fractal-like behaviour, which exists in many natural phenomena such as Brownian motion, turbulent flow, atmospheric pressure, the distribution of stars and the activity of the stock market [13]-[15], which are much better characterised by fractal geometry theory than by Euclidean geometry.

The rest of this paper is organised as follows. Section II discusses fractal models, Section III describes the experimental setup for image steganography, Section IV presents the experimental results detailing distortions on the cover images of different sizes, as well as fractal modelling of image distortion patterns, and, finally, Section V concludes this work.

II. FRACTAL COMPUTING

Fractal theory has been successfully used to depict many natural phenomena (for instance, random records in time) and more complex patterns associated with scientific phenomena [14], by characterising the structure embedded in one-, two- and three-dimensional space using fractal dimensions [15]. With the aid of fractal theory, some very

 ISBN: 978-1-61208-158-8

complicated phenomena such as dendrites and chaos can be handled by fractal mathematical functions, which describe the dependence of different features on the various parameters (e.g., fractal dimensions). It is well established that fractal theory provides a basis for studying irregular sets by modelling and making predictions.

Applications of fractal modelling to Internet systems [16]-[18] led to the elucidation of the likely cause of web-based traffic that has implications for the performance and stability of web applications. Fractal mathematics was used to simulate the growth of biological systems [19][20]. Fractal methods were employed to gain an understanding of the behaviour of users in a multipoint, interactive communication scenario, particularly the dynamics of user participation at a session level [21].

Fractal random walk (Fractional Brownian motion) was proposed as a model for the TCP packet stream [17], although the FBM model greatly overestimated the loss probabilities in the operating regions of interest, and the assumptions of the model did not appear to be satisfied by the two data traces that were examined [16]. A mathematical investigation of the accuracy of this approximation was then needed. More recently, fractal statistical models have been using for predicting web data traffic patterns, and the preliminary experimental results are exciting [22].

Recently, fractal methods were used for modelling and coding of residuals for excitation in the linear predictive coding of speech [23]. The research shows fractal modelling reduced the bit-rate while maintaining quality; a 6 kbps speech coder was implemented using the piecewise self-affined fractal model. Fractal image compression techniques have been employed in the production of steganographic image files [24]. Fractal image compression encoded images at low bit-rate with acceptable image quality, but time taken for encoding was large. Muruganandham et al. proposed a fast fractal encoding using particle swarm optimisation (PSO) that optimises MSE between range block and domain block whereas preserving image quality [25].

In this work, fractal models are suggested to elucidate the likely effects of the size of the cover image associated with different data embedding rates on image distortion patterns in image steganography.

III. EXPERIMENTAL SETUP

Least significant bit (LSB) insertion is one of the methods of embedding secret information in an 'innocent' cover image. For instance, 10101001 is an 8-bit binary number. The last bit in the binary number is the least significant bit because changing it has the nominal effect on the value. The method is based on the fact that change in the LSB information of some area of the image will not be noticeable by the naked eye.

StegaImage is a type of computer software capable of hiding secret information within a cover image [26]. The cover image should be a bitmap image, in which the file to be hidden is embedded. The stego image, consisting of the cover image and the secret file, is indistinguishable from the original cover image. By using the LSB insertion method, the software embeds the secret file in a 24-bit bitmap image. Fig. 1 shows the interface of StegaImage.

It is worth mentioning the 'Encrypt' and 'Decrypt' buttons at the interface should be renamed as 'Data embedding' and 'Data extraction', respectively, so as to avoid any misunderstandings.

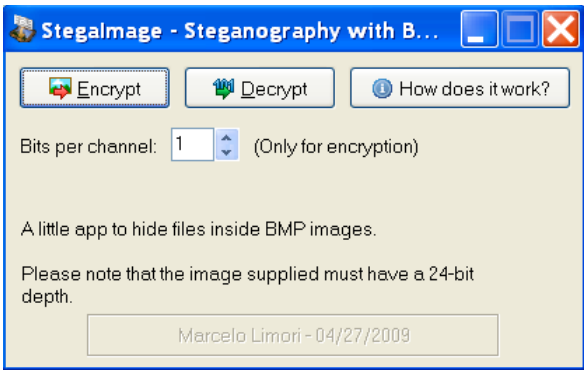


Figure 1. Steganographic software.

In a 24-bit bitmap, each pixel is represented as 3 bytes – one for red, one for green and one for blue, making up of a 24-bit number. Each colour is composed of 8-bit numbers, and the red, green and blue colours/channels create the final colour of the pixel. To hide something inside the image, software will replace the LSB of each 8-bit channel of every pixel, with the bits of the file to be hidden. This means the last bit in a byte can be overwritten without affecting the colour it appears to be. Therefore, the size of the cover image must be bigger than the size of the image to be hidden in order to accommodate in unrecognisable manner.

IV. RESULTS AND DISCUSSION

Three bitmap images with different sizes were used as cover objects in the experiments to hide smaller images, respectively. The sizes of the cover images vary from 772 KB, 1.34 MB to 1.56 MB.

A. Steganography in a 772 KB CoverImage

Fig. 2 shows the image to be hidden, the cover image (772 KB) and the stego images at various numbers of bits inserted into each channel (i.e. each pixel consisting of 24 bits).

A map image was used as the image to be hidden, which was embedded in the 772 KB bitmap image using StegaImage. No distortion was noticeable on the stego image when the map image was hidden in the cover image at 1 bit insertion per channel. However, inserting 5 or more bits per channel led to significant distortion to the original cover image.

In view of LSB data embedding, the cover image has the maximum data embedding capacity, depending on the

 ISBN: 978-1-61208-158-8

number of colours and palette in the cover image. With StegaImage, the bits of the image to be hidden were spread all over the least significant bits of the pixels of the cover image under a certain threshold of bits insertion per channel (here 5 bits per channel); when the number of bits inserted into a pixel was greater than the threshold, the distortion appeared on the upper side of the cover image due to the design principle of the software [26].

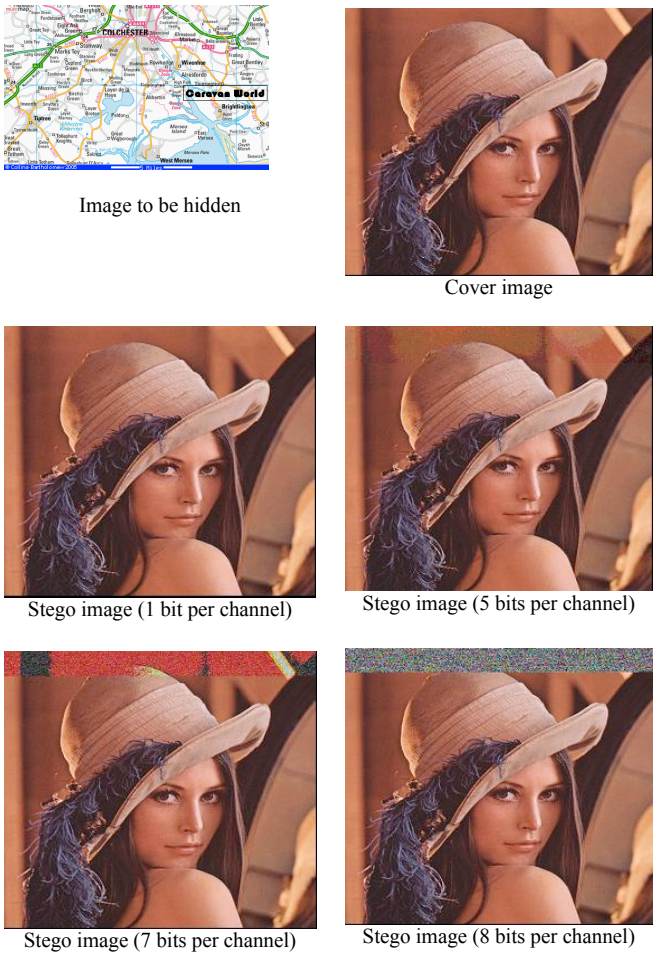


Figure 2. Steganography in a 772 KB cover image.

B. Steganography in a 1.34 MB Cover Image

Fig. 3 shows the map image to be hidden, the cover image (1.34 MB) and the stego images, each containing the cover image and the map image, as well as a distortion on the cover image when the map image was hidden in the cover image at up to 8 bit insertion per channel.

The StegaImage software uses last bits of each channel for insertion of data. If only the LSB was changed there was no effect on the cover image after data embedding; however, when the number of insertion bits per channel increased to 7, the change in the original cover image was perceptive, as shown on the last two stego images in Fig. 3.

C. Steganography in a 1.56 MB Cover Image

Fig. 4 shows the hidden image, the cover image (1.56 MB) and the stego images at various numbers of bits inserted into each channel using StegaImage software.

StegaImage was used to hide the picture of a girl inside the picture of a sport car. No difference between the cover image and the stego image could be seen by the naked eye if 5 bits per channel were replaced with the bits of the image to be hidden. But when the number of replaced bits per channel increased to 6, image distortion was clearly noticeable on the upper side of the cover image when compared with the original cover image.

Figure 3. Steganography in a 1.34 MB cover image.

D. Effect of the size of the image to be hidden

The StegaImage software that was used in our experiments to perform image steganography is based on LSB substitution, that is, the last bit in the binary number of the cover image is replaced with one bit of the image to be hidden. Such change in the LSB information of some area of the cover image will not be noticeable by the naked eye. So

 ISBN: 978-1-61208-158-8

the size of the image to be hidden must be smaller than the size of the cover image.

Our on-going research has been taking into account the size of the image to be hidden. It is anticipated that if the image to be hidden is very small in relation to the cover image, the hiding process may not produce a noticeable distortion in the cover image, depending on the ratio of the size of the image to be hidden to the size of the cover image, as well as the number of colours of the cover image.

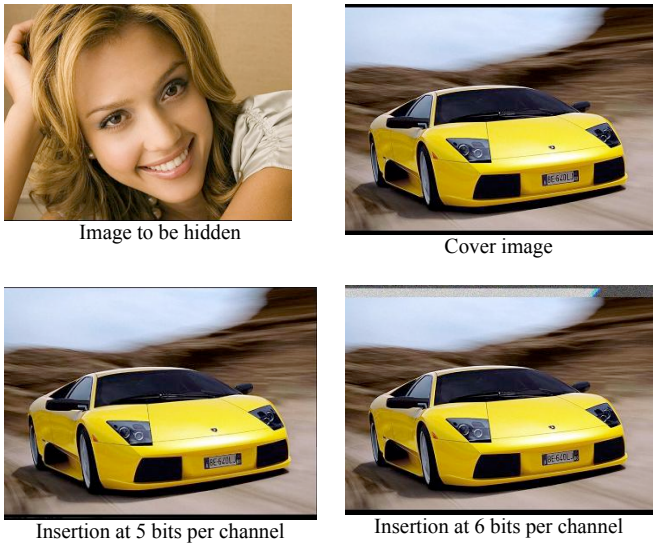


Figure 4. Steganography in a 1.56 MB cover image.

E. *Fractal Modelling of Image Distortion Patterns*

The maximum number of bits per channel that can be replaced without causing image distortion is a measure of the threshold of image distortion, which can be used to estimate the data embedding capacity. Table I lists the thresholds of image distortion for the three steganography experiments detailed in the previous sections.

TABLE I. THRESHOLD OF IMAGE DISTORTION

Size of cover image	Threshold
772 KB	5 bits / channel
1.34 MB	7 bits / channel
1.56 MB	6 bits / channel

To simulate the relationship between the threshold of image distortion and the size of the cover image, we used the successive random addition algorithm introduced by Voss [27] to generate fractal random walk simulation.

Fractal random walk simulation has been used for characterising random fractals [14], which may seem analogous to the threshold of image distortion vs. the size of the cover image, as randomness is inherent in most phenomena. A sequence of increments for random walk, i.e. increase in the threshold of image distortion in image steganography, can be generated from a sequence of Gaussian random variables, and the normalised variance of increments (V) is a function of the parameter t [15], $V(t) = |t|^{2H}$, where H is the Hurst parameter. Thus, the plot of $V(t)$ against t on a log-log plot has a slope, which is an estimation of $2H$.

In this case, $V(z)$ represents the fluctuation of the threshold of image distortion in image steganography and is a function of the size of the cover image (z), given by

$$V(z) \propto |z|^{2H} \qquad (1)$$

which is fitted with a range of statistic equations, and z is the size of the cover image used to embed the image to be hidden. The value of the Hurst parameter H will characterise the degree of burstiness of image distortion.

Fig. 5 shows the threshold of image distortion as a function of the size of the cover image based on the steganography experimental results listed in Table I. A fractal Hurst parameter of $H = 3.1973 / 2 = 1.5987$ was obtained from the slope of log (Threshold) vs. log (Size of cover image) according to (1).

According to random fractal theory, the value of the Hurst parameter should be between 1 and 2. Giving that the experimental value of H is equal to 1.5987, it indicates that image distortion has fractal-like behaviour in terms of the threshold of bits insertion per channel as a function of the size of the cover image. The Hurst parameter would maintain on many or all scales, so its value could be then used to describe the dependence of the image distortion threshold on the size of the cover image, even for a very large cover image, e.g., 100MB in size.

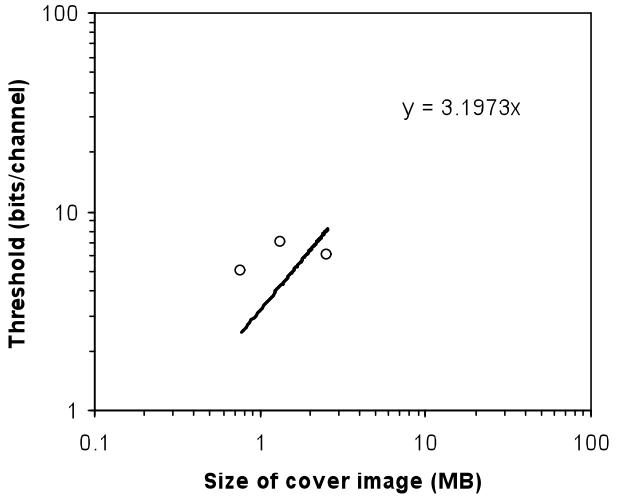


Figure 5. Threshold of image distortion as a function of the size of the cover image.

V. CONCLUSION AND FUTURE WORKS

In this paper, we have applied fractal computing to modelling of image distortion patterns in image steganography. The successive random addition algorithm has been used to simulate the image distortion patterns of embedding the image to be hidden in the cover image. A distortion pattern is beginning to emerge from our analysis of the steganography experimental data. The distortion pattern identified in this work could make predictions of future trends.

Other fractal methods such as R/S scaling and Fractal random walk could be used to simulate the image distortion patterns of image steganography in the future. Comparisons in the fractal Hurst parameters obtained from different fractal models would be able to further confirm the fractal behaviour of image distortion patterns in image steganography. In addition, investigation into the effect of the size of the image to be hidden on image distortion patterns will also be the subject of future work.

ACKNOWLEDGMENT

The authors would like to thank M.K.M. Jayawardena, G. S. Gayan De Fonseka, and B. Abhishek for providing some images, and thank Marcelo Limori for developing the StegaImage steganographic tool.

This work was supported in part by the British Government (Grant ktp8263).

REFERENCES

[1] D. Artz, "Digital Steganography: Hiding data within data," IEEE Internet Computing, pp. 75-80, June 2001.

[2] J. E. Wingate, "Digital Steganography: An introduction to the practice of digital information hiding," Digital Forensics Magazine, pp. 73-76, June 2010.

[3] F. A. P. Peticolas, R. J. Anderson, and M. G. Kuhn, "Information Hiding – A Survey," IEEE Trans. Proc. Thy., vol. 87, no. 7, pp. 1062-1078, 1999.

[4] P. Bao, and X. Ma, "MP3-resistant music steganography based on dynamic range transform," Proc. IEEE International Symposium on Intelligent Signal Processing and Communication Systems, Seoul, Korea, Nov. 2004, pp. 266-271.

[5] N. F. Johnson, Z. Duric, and S. Jajodia, Information hiding: Steganography and watermarking-attacks and countermeasures, Kluwer Acade. Publishers, 2000.

[6] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB Steganography in Colour and Grayscale Images," IEEE Multimedia, vol. 8, pp. 22-28, 2001.

[7] S. Dumitrescu, X. Wu, and Z. Wang, "Detection of LSB Steganography via Sample Pair Analysis," IEEE Transactions on Signal Processing, vol. 51, no. 7, pp. 1995-2007, 2003.

[8] S. Lyu, and H. Farid, "Detecting hidden messages using higher-order statistics and support vector machines," Lecture Notes in Computer Science, vol. 2578, pp. 340-354, 2003.

[9] Y. Miche, B. Roue, A. Lendasse, and P. Bas, "A feature selection methodology for steganalysis," Proc. International Workshop on Multimedia Content Representation, Classification and Security, Istanbul, Turkey, September 2006, pp. 49-56.

[10] L. D. Olsen, and D. Delen, Advanced Data Mining Techniques, Springer, USA, 2008.

[11] S. Ahmed, P. Pan, and S. Tang, "Clustering websites using a MapReduce programming model," Journal of Communication and Computer, USA, vol. 7, no. 9, pp. 18-26, 2010.

[12] R. S. Ahmed, "Applications of Data Mining in Retail Business," in Information Technology: Coding and Computing, vol. 2, 2004, pp. 455-459.

[13] J. Feder, Fractals, Plenum Press, New York, 1988.

[14] K. Falconer, Fractal Geometry: Mathematical Foundations and Applications, John Wiley & Sons, Chichester, 1990, pp.146-160.

[15] B. H. Kaye, A Random Walk Through Fractal Dimensions, VCH, Weinheim, 1994, pp. 179-188.

[16] D. Heyamn, "Some issues in performance modelling of data teletraffic," Performance Evaluation, vol. 34, pp. 227-247, 1998.

[17] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of Ethernet traffic," IEEE/ACM Trans. Networking, vol. 2, pp. 1-15, 1994.

[18] K. Rezaul, S. Tang, T. Wang, and A. Paksta, "Empirical distribution function test for world wide web traffic," Proc. IADIS International Conference Applied Computing 2004, Lisbon, Portugal, March 2004.

[19] S. Tang, Y. Ma, and I.M. Sebastine, "The fractal nature of Escherichia coli biological flocs," Biointerfaces, vol. 20, pp. 211-218, 2001.

[20] S. Tang, "Computer simulation of fractal structure of flocs," Encyclopaedia of Surface and Colloid Science, pp. 1162-1168, August 2006. Taylor & Francis, ISBN: 978-0-8493-9615-1.

[21] S. Bhatti, "Modelling user behaviour in networked games," Proc. the ninth ACM international conference on Multimedia, Ottawa, Canada, 2001, pp. 212 - 220.

[22] S. Tang, and H. Kazemian, "Simulation of web data traffic patterns using fractal statistical modelling," Lecture Notes in Computer Science, Springer, in press, 2011.

[23] W. Kinsner, and E. Vera, "Fractal modelling of residues in linear predictive coding of speech," Proc. 8th IEEE International Conference on Cognitive Informatics, 2009, pp. 181-187.

[24] P. Davern, and M. Scott, "Fractal based image steganography," Lecture Notes in Computer Science, vol. 1174, pp. 279-294, 1996.

[25] A. Muruganandham, and R. S. D. Wahida Banu, "Effective MSE optimization in fractal image compression," International Journal of Computer Science and Information Security, vol. 8, no. 2, 2010.

[26] Computerworld. Steganography: Hidden Data. Quick study by Deborah Radcliff. [online] 2010. Available at http://www.computerworld.com/securitytopics/security/story/0,10801,71726,00.html.

[27] R. F. Voss, "Random fractal forgeries," in Fundamental Algorithms for Computer Graphics, vol. 17, R. A. Earnshaw, Eds, NATO ASI Series F, Computer and System Sciences, 1985.

 ISBN: 978-1-61208-158-8

Reducing User Error by Establishing Encryption Patterns

Anthony Gabrielson, Haim Levkowitz
Department of Computer Science
University of Massachusetts Lowell
One University Avenue, Lowell, MA 01854, United States of America
agabriel@cs.uml.edu
haim@cs.uml.edu

Abstract— **This paper is motivated by a desire to create a user friendly, technically flexible approach to cryptography. This approach can create a flexible design pattern that is useable under a wide variety of circumstances from broadcast media to existing network protocols. This is accomplished by designing a new, key piece of infrastructure used for cryptography key lookup. Once the key is obtained the application/protocol developer is given the flexibility to meet their requirements.**

Keywords-Security Patterns; Patterns of Trust; Authorization Patterns

I. INTRODUCTION

Encryption technology needs to evolve to a point where end-users are not aware they are using it [1]. To accomplish this goal a common infrastructure, which can be used across a wide spectrum of applications, is needed. Currently deployed technologies, namely Secure Sockets Layer (SSL) and Certificate Authorities, are no longer sufficient because in a quest for the lowest possible price the current infrastructure has shown it is extremely vulnerable to attacks due to woefully inadequate administrative procedures [2]-[3] and technical issues that prevent it from being used to solve an array of problems from email to media distribution.

This paper presents a solution that is extremely configurable and distributable. It develops a security pattern by leveraging existing technologies and ideas, and using them in a manner similar to that of other widely deployed technologies that make up the fundamental building blocks of the Internet. The structure of this solution also allows developers to re-examine other features in the TCP/IP suite, like UDP broadcast and multicast, enabling easier distribution of encrypted content to many users simultaneously.

The rest of the paper is organized as follows. The remainder of the introduction defines trust and requirements. The foundations for the general implementation of our pattern is described in Section II. Section III describes our concept of operations for a TCP and UDP implementation. The implementation and testing details for our pattern are described in Section IV. Sections V-VI describe the benefits and shortcomings of this pattern. Future improvements are described in Section VII. The conclusion follows in Section VIII.

A. Trust

The main goal of this research has been to establish a trusted encrypted channel between two points that is easily configurable by the developer while requiring minimal user input. The channel must not work if there is any breakdown in trust. To accomplish this we first need to define trust. A proper trust definition is essential to understanding requirements. This is the definition of trust we have been working with:

> *"Trust is a mental state comprising: expectancy — the [trusting individual] expects (hopes for) a specific behavior from the [entity s/he is putting trust in] (such as providing valid information or effectively performing cooperative actions); belief — the [trusting individual] believes that the expected behavior occurs, based on the evidence of the [trusted entity's competence] and goodwill; and willingness to take risk — [based on that belief] the [trusting individual] is willing to take risk [for the purpose of achieving some desired result.]"* [4]

B. Requirements

Our requirements are fairly straightforward. The proposed solution only requires a minimal amount of setup information to establish an encrypted channel. If there are any key or encrypted channel problems the recipient shall not be able to decrypt the data. The encryption algorithm needs to support encryption with multiple public keys. The implementation also needs to be able to support non-malleability [5], non-repudiation, and authentication.

The added interaction between the user and the client application required for cryptography needs to be minimal. We focus on an API that is used at the application layer of the ISO network model; our approach utilizes a separate key server, and requires a small amount of additional information, almost all of which can be automated, to the client application: 1. the location where the key server resides on the network; 2. a local username; and 3. the recipient's username and host. The key server details can be hidden by creating an additional user variable on the local system or creating a new DNS record type that specifically points to the key server. The local username is also typically available as a user variable. The recipient's username could also be standardized under certain circumstances with a *getservbyname()* [6] system call. This could reduce the

 ISBN: 978-1-61208-158-8

required user input to just one piece of information -- the destination host name, which is already required.

Cryptography software developers have largely ignored the TCP/IP broadcast and multicast features; however this important functionality needs to be considered [7]. To help facilitate those needs, copyright holders increasingly need to be able to easily distribute intellectual property while ensuring it is protected. Software should be able to encrypt the same data with multiple public keys and transmit it such that only the intended recipients will be able to decrypt it. This capability, combined with an understanding of the network media reliability allows broadcast and multicast features to be utilized with cryptography.

Our proposed solution also needs to support non-repudiation and authentication; of these two, the more important one is non-repudiation: It is extremely important that data sources cannot be disavowed. While key-based authentication is desirable it may be appropriate in some situations to add an additional level of trust; an application developer may wish to also add a username and a password or a secondary key file, such as a crafted image, that will only be accepted from a designated user.

II. PROPOSED SOLUTION

A. General Encryption Technology

Currently, only the PGP/GPG cryptography standard meets the encryption requirements of this project. However, it does have a few trade-offs that must be handled. One major trade-off is the zip compression algorithm utilized by the PGP standard, as illustrated in Figure 1. Since PGP uses compression, a particular data size to cipher text size cannot be guaranteed. We address this later in this paper.

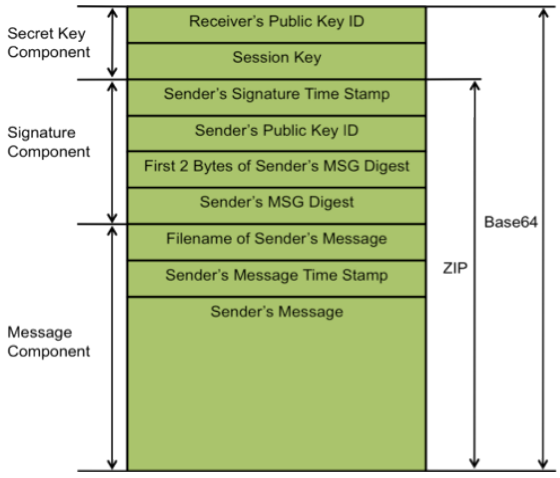


Figure 1: PGP Data Shape [8]

B. General Terms

There are three terms that will appear throughout the remainder of this paper – *Originator, Destination*, and *Encryption Key System (EKS).* All data and connections originate from an Originator, which could be a Client or a Server depending on protocol configuration. The Encryption Key System (EKS) serves public cryptography keys (Pk) to the Originator. It works similar to the Domain Name System (DNS) [9] except that rather than providing IP addresses it will provide public keys.

C. Web of Trust Model

Lucas tells us that PGP goes out of its way not to define trust [10], which leads to an application that is difficult to use properly [1]. A trust (in performance and belief) model needs to be established that automatically finds and trusts keys. The trust in performance is based on the destination's ability to pass a challenge by decrypting the received data. The trust in belief is in the process of receiving the correct public key in the process. If both a trust in performance and belief are satisfied then public keys should be fully trusted.

Huang has defined a semantic for a belief relationship [4]:

$$\tau\beta(\delta,\varepsilon,\xi,\kappa) \equiv \alpha(\varepsilon, \kappa \rightarrow \xi) \supset \psi(\delta,\kappa \rightarrow \xi) \qquad (1)$$

where a trust-in-belief relationship, $\tau\beta(\delta,\varepsilon,\xi,\kappa)$, represents that trusting-entity δ trusts trusted-entity ε regarding ε*'s* belief ξ in context κ. This trust relationship means that if ε believes ξ in context κ, then δ also believes ξ in that context [4].

A semantic for a performance relationship can be formulated as [4]:

$$\tau\Pi(\delta,\varepsilon,\xi,\kappa) \equiv \alpha(\varepsilon, \kappa \rightarrow \xi) \supset \psi(\delta,\kappa \rightarrow \xi) \qquad (2)$$

where a trust-in-performance relationship, $\tau\Pi(\delta,\varepsilon,\xi,\kappa)$, represents that trusting-entity δ trusts trusted-entity ε regarding ε*'s* performance ξ in context κ. This relationship means that if ε makes ξ in context κ, then δ believes ξ in that context [4].

Trust can now be directly defined such that:

$$\tau(\delta,\varepsilon,\xi,\kappa) \equiv \tau B(\delta,\varepsilon,\xi,\kappa) \cap \tau\Pi(\delta,\varepsilon,\xi,\kappa) \qquad (3)$$

That is to say that if the destination believes it is communicating with the correct originator and that originator is performing as expected it would be trusted.

D. Web of Trust in Practice

In practice, the trust relationship begins with the local EKS Server. Figure 2Figure 2 illustrates the communication that might be required for an originator to obtain a particular key that is not hosted on a Local EKS server. If the Pk were hosted locally the Local EKS Server would simply return it. However in this case the originator will connect to the Local EKS Server that will locate and provide the Originator with the desired Pk.

 ISBN: 978-1-61208-158-8

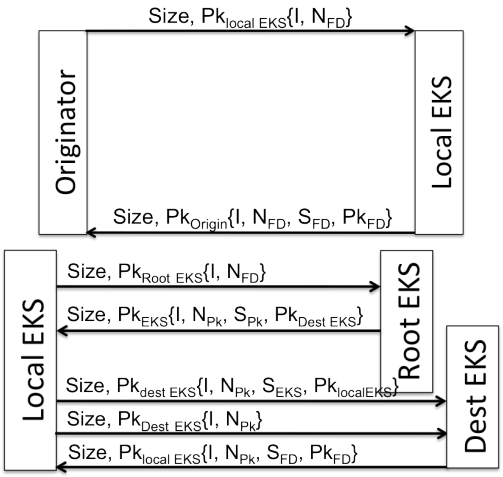


Figure 2: Web of Trust

The Local EKS Server starts this process by communicating with the appropriate Root EKS Server that will provide the Local EKS Server with $Pk_{Dest\ EKS}$. The Local EKS Server will then contact the Destination EKS Server asking for Pk_{FD} or Final Destination; the Final Destination is not illustrated in the figure. Since the Destination EKS Server does not have $Pk_{local\ EKS}$ it is sent during the initial connection. The process of looking up a Pk could best be compared with looking up an IP address. Once the IP address has been obtained the originator can begin to establish communication with the destination host.

The Originator trusts the key provided by the Local EKS server because of the process required to retrieve it. The Originator and Local EKS Server have a priori knowledge of their Pks. The Local EKS Server has the same a priori knowledge with the Root EKS Server. The Local EKS Server only needs to send its Pk to the Destination EKS Server. This process has three main strengths 1. it only exposes the ability to encrypt a message that can only be decrypted by the destination; 2. IP addresses can still change since key lookup is based on name; 3. if a Destination's IP address is spoofed the Local EKS Server will not have access to the spoofed Destination's Pk at lookup; so the spoofed Destination will be unable to decrypt any received data. The Originator trusts the received Pk because those three characteristics create a trust in belief and performance.

III. Concept of Operation

A. TCP Example

Let us walk through the implementation of an FTP like protocol using TCP. The Originator, in this case the Client, sends a request to the EKS Server asking for the Destination, or Server's, Pk. The EKS Server responds with the Destination's key name (N_{Pk}) and it's Pk_{Server} – all encrypted with the Pk_{client}. The Originator is now ready to initiate encrypted communication with the Destination. The Client opens a connection and sends an encrypted IDENT message type that includes the Pk_{client}. The Destination then decrypts and installs Pk_{client} and establishes trust in it. A two-way encrypted communication channel between the Client and the Server is now established. The Client follows up on its initial connection with a request in field I, in the second communication between the client and server; since the Server has installed and trusts Pk_{client}, which is the first communication between the client and server, it is able to respond as illustrated in Figure 3***Error! Reference source not found.***.

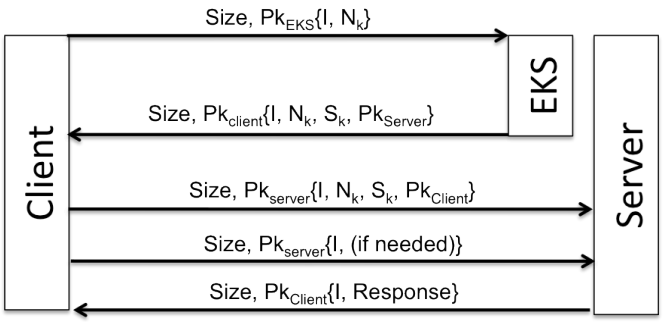


Figure 3: TCP Example

B. UDP Example

Let us now walk through a protocol that implements the UDP broadcast capability. The Originator or Server in this case, sends a request to the EKS Server asking for the Destination, or Client's, Pk. The EKS Server responds with the Destination's N_{Pk} and Pk_{client} – all encrypted with Pk_{server}. The Originator installs Pk_{client} and is now ready to establish an encrypted one-way channel with the Destination as illustrated in Figure 4**Error! Reference source not found.**. Since the Client will never need to reply in this scenario, the Server does not provide Pk_{server} to the Client.

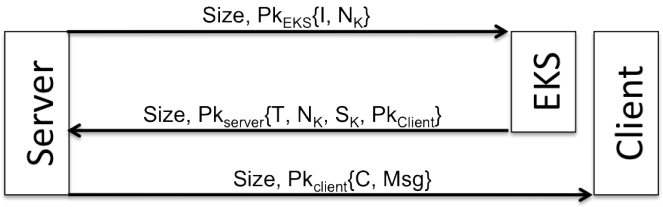


Figure 4: UDP Example

To work within UDP limitations it is recommended that data be parsed such that its encrypted size is less then or equal to the size of the Maximum Transmission Unit (MTU) [11] of the particular network medium. A counter has also been inserted to enhance the resolution of the PGP time stamp illustrated in Figure 1Figure 1 to better enable any data buffering implemented at the destination.

IV. Implementation and Testing

A. Initial Setup

A few steps are necessary to set up the infrastructure required by the EKS Server, Originator, and the Destination. While these steps may seem in depth this process can and

should be automated for a final solution. First, all three accounts must initialize PGP keys.

The EKS Server must have both $Pk_{Destination}$ and $Pk_{Originator}$ installed and trusted in its key ring. The Originator must initially have and trust $Pk_{EKS\ Server}$. The EKS Server will provide $Pk_{Destination}$ upon request. After the EKS Server transfers $Pk_{Destination}$ the Originator will install and trust it. The Destination does not initially need any Pks installed in its key ring because it will not initiate contact with any other host. When the Originator initially connects $Pk_{Originator}$ will be transferred during the initial message if needed. The Destination will trust and install $Pk_{Originator}$ if it is transferred.

B. Implementation

We have implemented this solution architecture using both the 32- and 64-bit versions of the Ubuntu Linux 10.10 operating system hosted on VMware Workstation and Fusion; we see no reason this solution will not work on other platforms like Microsoft Windows or Apple's Macintosh OS X. GPG version 1.4.10 was used and installed using the apt tool. We used Qt 4.7, a cross-platform application framework, to construct these two tools together. We implemented the TCP and UDP communication protocols. This solution makes communication with a spoofed IP address impossible because an attacker will not have access to the required secret keys (Sk) $Sk_{Originator}$ or $Sk_{Destination}$ to decrypt the transmitted data.

C. Testing

There can be many use cases for this research; we tested two of them: TCP, and UDP Broadcast Communication. In both cases all three hosts were on the same local network. The EKS Server trusted Pk_{EKS}, $Pk_{Destination}$, and $Pk_{Originator}$. The Originator trusted two public keys before running – its own and Pk_{EKS}; after connecting to the EKS Server the Originator also trusted $Pk_{Destination}$. The Destination initially trusted one key – its own; after the Originator connected to the Destination it trusted $Pk_{Originator}$ if transmitted.

D. GPG Commands

A small number of `GPG` commands were needed for this implementation [12] and we will highlight a few of the more compelling features. The `encrypt` capability includes the –s and –r options. The –s option signs the data with the sender's key, which is helpful for TCP communication; assuming the Destination(s) are familiar with the Originator, this can be used for sender verification since the recipient will understand the sender's signature. The –r option allows users to be added to the encryption stream and can be called multiple times with different usernames enabling a UDP broadcast capability where only the intended recipients will be able to decrypt the stream.

V. BENEFITS

The implementation described provides several key benefits to existing solutions. It enables non-repudiation and authentication through the built-in PGP capabilities and a defined trust relationship. The user provides minimal input to start and maintain an encrypted channel, thus simplifying the end-user's experience [1] by providing fewer opportunities to make a mistake. Our solution is more scalable than other key distribution techniques as explained below. Finally, communication is task-based, allowing better protection of information, such as usernames and/or passwords, because the API is used at the Application Layer.

Since PGP enables non-repudiation and authentication and we have established an automated trust definition users can connect to a host and trust that they have in fact connected to the right host because the EKS Servers are trusted and match DNS results. Our defined trust relationship may in some cases delude the trust PGP originally intended so usernames and passwords can be used additionally and linked to a particular key signature, increasing the overall security.

The user is never asked any security questions, unless the implementation utilizes the added security of a username/password. If a particular host is unable to communicate it may indicate that an attack is underway and communication at this point in time is not desirable. In this case a malicious host will not be able to decrypt the data since it does not posses the correct private key required to decrypt the data. The method to download new public keys establishes trust and there is no way to compromise data integrity. This feature prevents attack vectors like evilgrade [13], presented by Amato, since there is no way to decrypt the transmitted network traffic.

The EKS server and protocol design have several benefits when compared to a Kerberos Key Distribution Center [14] implementation. The two primary differences are: first, EKS servers can be scaled up to cache keys and thus reduce the expense of continuously looking up the same keys, similar to DNS capabilities. The other primary difference is that the EKS server does not need to contact both the Originator and Destination with a ticket since the Originator will send the Destination its key during the initial connection if needed. The added capabilities of this solution allow it to be more scalable than the alternative.

Protocols can become more task-oriented, enabling better sandboxing [15]. Since each task or daemon can use its own private key, tasks can be better partitioned on the same system. Information, like usernames and/or passwords, can be encrypted for transit so that it cannot be sniffed [16], which will eliminate the Man-in-the-Middle Attack [17]. Data can be stored in the cloud in a manner that the service provider, or any attacker, will not have access to it.

VI. SHORTCOMINGS

Our proof-of-concept exhibits two main shortcomings. All Originator traffic must start with its home EKS Server. The current implementation is performance-limited due to file IO; GPG requires that all encryption and decryption activity go through a file.

All Originator communication must start with its EKS Server because of the trust relationship between the two. If users are mobile and using random access points to access the Internet their home EKS Server must be publicly available on the Internet. This only poses a minimal security

 ISBN: 978-1-61208-158-8

risk since the EKS Server is a low value target since it only hosts and transmits public keys.

The current GPG implementation is extremely file IO dependent; all communications are realized through a file on both sides of the connection. This can have an impact on the speed of the communications. This was observed during early testing on a Windows XP system, where a noticeable delay was perceptible. The remainder of the implementation and testing was done on a Linux system in a virtual machine and file IO was not a perceptible slow down, however simultaneous connections were limited. We currently view this limitation as minimal since it could easily be resolved with a GPG API.

VII. FUTURE IMPROVEMENTS

This proof-of-concept still needs improvements. First and foremost our research has so far focused on two use cases and more are needed to address other scenarios. In addition, there are several other areas that also need improvement, such as, key caching time limits, and a GPG API to avoid file thrashing.

More use cases need to be added to the EKS Server. There are two main EKS Server use cases -- when the Originator and Destination are on the same network and when they are further apart on the Internet. Additional protocols need to be implemented for the second use case since the local EKS Server will need to be able to communicate with other EKS Servers to look up other public keys.

After more use cases are added the EKS Server needs a time based cache table added for non-local hosts. Similar to DNS [9], an EKS Server should cache non-local Pk for a certain amount of time to reduce network traffic and CPU utilization. This feature is also a security enhancement since key pairs could be allowed to sunset after a particular period of time.

This implementation could also be considerably quicker with GPG provided API level access; this would eliminate almost all file IO transactions. A linkable library that provides the same capability but provides byte level access to data input and output would accomplish this.

VIII. SUMMARY

The solution proposed in this paper forms the foundation of a new cryptography design pattern. This design pattern allows developers the flexibility they require and users the security and simplicity they need to accomplish a task. This pattern is only possible because of a proper trust definition that enables security issues to be handled automatically.

Modern computing is already complex enough without adding more layers that users need to worry about. Application developers need secure flexible cryptography libraries that implement best practices that are easy to use to solve encryption problems from media distribution to email. Encryption technology needs to become ubiquitous in computing. To create this capability the solution needs to be as close to invisible as possible to the end user; the user must be able to focus on the task at hand. While the user is focusing on her/his task any security-related questions necessary to achieve these goals must be automatable to avoid overwhelming the user.

To become more ubiquitous the solution needs to be flexible enough to enable as many existing data distribution capabilities as possible while restricting decryption privileges to only the intended recipients. Once this goal is achieved many of the existing problems currently being experienced will be obsolete.

REFERENCES

[1] A. Whitten. (1999). Why Johnny can't encrypt: a usability evaluation of PGP 5.0. In SSYM'99 Proceedings of the 8th conference on USENIX Security Symposium, ACM, Ed. Vol. 8. 15.

[2] E. Nakashima. (2011) Washington Post webpage on Cyberattack. [Online]. Retrieved June 17, 2011, From The Washington Post: http://www.washingtonpost.com/world/us_agencies_respond_to_cyberattack_on_information_security_firm/2011/03/23/ABDhjoKB_story.html?wprss=rss_homepage

[3] F. Rashid. (2011) eWeek webpage on Fake SSL certificates, [Online]. Retrieved June 17, 2011, From eWeek.com: http://www.eweek.com/c/a/Security/Fake-SSL-Certificate-Incident-Highlights-Flaws-in-DNS-Comodo-CEO-440985/

[4] J. Huang. (2010). A formal-semantics-based calculus of trust. In Internet Computing, IEEE. iEEE, 38 – 46.

[5] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. (1998). Relations Among Notions of Security for Public-Key Encryption Schemes. In Advances in Cryptology – Crpto '98.

[6] FreeBSD Man Page., [Online]. Retrieved June 17, 2011, From The UNIX.com Man Pages: http://www.unix.com/man-page/FreeBSD/3/getservbyname/

[7] Lixin Gao; Towsley, D. ; , "Threshold-based multicast for continuous media delivery," *Multimedia, IEEE Transactions on* , vol. 3, no. 4, pp. 405-414, Dec 2001

[8] J. Wang. (2009). Computer Network Security Theory and Practice. Springer.

[9] Albitz, Paul. 2001. DNS and BIND, 4th Edition. Oreilly.

[10] M. Lucas. (2006). PGP & GPG Email for the practical paranoid. No Starch Press.

[11] Reviriego, P. ; Sanchez-Macian, A. ; Maestro, J.A. ; Bleakley, C.J. ; , "Increasing the MTU size for Energy Efficiency in Ethernet," *Signals and Systems Conference (ISSC 2010), IET Irish* , vol., no., pp. 124-128, 23-24 June 2010

[12] gnupg: Documentation, [Online]. Retrieved June 17, 2011, From The GNU Privacy Guard: http://www.gnupg.org/documentation/manuals/gnupg-devel/GPG-Configuration-Options.html#GPG-Configuration-Options.

[13] F. Amato. Evilgrade: you have pending upgrades... . Ekoparty Security Conference, Buenos Aires, Argentina, Nov 30 – Dec 1, 2007.

[14] Neuman, B.C.; Ts'o, T.; , "Kerberos: an authentication service for computer networks," *Communications Magazine, IEEE* , vol. 32, no. 9, pp. 33-38, Sep 1994.

[15] Greamo, C.; Ghosh, A.; , "Sandboxing and Virtualization: Modern Tools for Combating Malware," *Security & Privacy, IEEE* , vol.9, no. 2, pp. 79-82, March-April 2011.

[16] Qadeer, M.A.; Zahid, M.; Iqbal, A.; Siddiqui, M.R.; , "Network Traffic Analysis and Intrusion Detection Using Packet Sniffer," *Communication Software and Networks, 2010. ICCSN '10. Second International Conference on* , vol., no., pp. 313-317, 26-28 Feb. 2010

[17] M. Marlinspike. (2009). Defeating SSL. In Black Hat DC.

 ISBN: 978-1-61208-158-8

Application-Domain Classification for Security Patterns

Michaela Bunke, Rainer Koschke, and Karsten Sohr
Center for Computing Technologies (TZI),
Universität Bremen, 28359 Bremen, Germany
{mbunke|koschke|sohr}@tzi.de

***Abstract*—Security patterns are best practices to handle recurring security problems. Existing classifications for security patterns consider only a small number of patterns, and their purpose is often focused on implementations issues. Therefore we identify missing aspects in existing classifications and introduce a new classification scheme based on application domains. This scheme is based on a literature survey on security patterns published in the period of 1997 to 2010 to cover the whole bandwidth of exiting security pattern.**

***Index Terms*—Security Patterns.**

I. Introduction

Software security is an emerging area in software development. More and more vulnerabilities are published and compromise systems and their users [1]. Software designers and programmers are therefore faced with applying security solutions to software systems. In the domain of software development, design patterns have been proposed as specific solutions for recurring problems in software design [2].

Yoder and Barcalow summarized some existing patterns targeting security and introduced the term *security pattern* [3], only three years after Gamma et al. [2] proposed their design patterns. Security patterns are best practices aiming at ensuring security [4], [5].

Existing security pattern classifications are often based on a few security patterns. Their scope is often limited to special areas such as implementation patterns. For instance, Hafiz et al. formed their classification with only 14 security patterns [6], but there exist many more security patterns.

Therefore, we conducted a systematic literature review and collected the published security patterns in the period of 1997 to 2010. We propose a new classification scheme that summarizes 409 security patterns, a much longer list than the one by Yoder and Barcalow [3]. Moreover, our classification scheme supports the selection by application domain, which is relevant for researchers and practitioners who are interested in security patterns for domain-driven tasks.

Our motivation conducting this literature research and shaping a new classification was to get an overview on existing patterns and organize them in application domains. The research focus of our future research is to detect and validate software security patterns implemented in code. To get started, we need a clear picture of what kinds of security patterns exist.

The remainder of this paper is structured as follows. An overview of classifications in general is given in Section II. Existing classification approaches for security patterns will be described in Section III. In Section IV, we describe our literature survey and will introduce our new classification. Afterwards, we will conclude in Section V.

II. Requirements for Classifications

The increasing number of patterns makes it necessary to develop classifications. This sections describes general requirements for classifications in general and on security patterns in particular.

A classification should be based on systematic methodologies and techniques to organize a mass of patterns. A classification organizes patterns into groups of patterns that share one or many properties such as the application domain or a particular purpose. The kind of properties that should be used is not fixed. A pattern can have more than one specific property. Therefore, it may be included in more than one classification category.

According to Buschmann et al., a pattern classification scheme should meet some basic properties [7]. It must too be simple and easy to learn. This should be supported by using only few classification criteria to reduce the complexity and ambiguity for users. In addition, a classification should reflect the main properties of a pattern to classify it. Last but not least, a classification scheme should provide the possibility to classify new patterns.

Fernandez et al. pointed out that a classification should make the application of patterns much easier along the software lifecycle [8]. Due to the fact that it is impractical to look at details of a pattern to pick the right pattern for the problem at hand, a classification should help to understand the essential nature and value of patterns.

A simple and intuitive classification is shown in Figure 1. A natural way to classify pattern is to categorize them according to the discipline where they are applied. Patterns can also be distinguished by their application domain like network, embedded systems or distributed systems. Another way to differentiate patterns is to determine their programming concepts paradigms such as object-oriented or imperative programming language concepts. Moreover, it is possible to categorize them depending on the level of abstraction they address, e.g., design or coding patterns. In addition, the criterion *purpose* represents the kind of problem a pattern solves and when it may be applied.

 ISBN: 978-1-61208-158-8

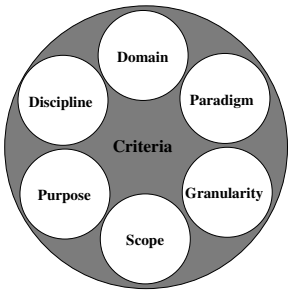


Fig. 1. Intuitive classification

III. Existing Classifications

The presented classification criteria in Section II are simple, but do not always fit for selecting the right pattern for a special purpose because of their generality. Therefore more specific classification schemata based on one or more criteria have been developed to meet special purposes like the *structural* and *behavioural* distinction on object creation and usage by Gamma et al. [2].

One of the simplest classifications for security patterns was used by Kienzle et al. [9]. They presented the *structural* and *procedural* criteria for the differentiation of the patterns described in their final report. If a security pattern is concerned about compositions or structures that are implemented in a final software product, it is *structural*. Otherwise a security pattern is *procedural*.

Konrad et al. [10] proposed a classification method for security patterns re-using the classification for design patterns such as *creational*, *structural* and *behavioural* from Gamma et al. [2]. They enhanced their classification by adding further categories such as network, host, and application. In their work, they considered only the security patterns introduced by Yoder and Barcalow [3].

Schumacher's security patterns book offers a new classification system [11]. The classification is based on Zachman's framework [12] for enterprise architecture. It is presented along two dimensions. One dimension represents different views on the interrogatives "what", "how", "where", "who", "when", and "why". The second dimension shows different information model views such as *business model* or *technology model*. Schumacher et al. enhanced this framework by adding the column *security* to emphasize the security view and to be able to address all model levels. They organized only the patterns contained in the book into their classification.

According to the Java Platform, Enterprise Edition (JEE) pattern classification by Alur et al. [13], Steel et al. [14] classify their JEE security patterns in a similar way. They separate their patterns in layers that are typical for the development in the JEE domain such as *Web*, *Business*, *Web Service* and added a fourth tier that represents the special issue of *identity management*. This classification is only designed for the special purpose of JEE patterns and does not consider other types of patterns.

Rosado et al. related security requirements to security patterns and classified security patterns into two categories: architectural and design patterns [15].

Hafiz et al. proposed a classification based on a tree structure combined with the STRIDE-Model [16] to join the software and security view in terms of security patterns [6]. The STRIDE-Model is normally used for threat modelling including identifying and prioritization of security vulnerabilities. They tested their classification with 14 different security patterns, and compared it against other available security classifications.

VanHilst et al. introduced a multi-dimensional matrix of concerns to classify security patterns [17]. It addresses the problem coverage and pattern classification. Their idea was that each matrix dimension represents a well-defined list of concerns, which is presented along one single axis. To classify security patterns, the primary dimension contains concerns of life-cycle activities like *domain analysis* or *requirements*. The second dimension differentiates security patterns by their component source type such as *new code*, *legacy*, or *wizard-code*. Other dimensions may hold types of security responses like prevention or mitigation, but they can also be further customized to a user's need. Their classification was tested by different members of their team, who added six different security patterns to the classification.

Fernandez et al. state that security patterns are architectural patterns [18]. Therefore, their approach deals with two classifications that differ in different viewpoints of security patterns. On the one hand, they introduced a classification by a hierarchy of layers and on the other hand, they proposed a classification based on the relationships between patterns by using an automatic relationship extraction and analysis technique. This classification is abstract and only regards a small number of security patterns.

Washizaki et al. point out that the previously introduced classifications have only a few dimensions and do not embrace the relations between patterns [19]. Therefore, they introduce a meta-model to express the patterns' properties and relations uniformly. The base is an excerpt of the multidimensional classification dimensions presented by VanHilst et al. [17]. They picked out the dimensions as follows: *Lifecycle stage*, *Architectural level*, *Concern*, *Domain*, *Type of pattern* and *Constraint*. In addition, they used the three UML standard relationship types *association*, *generalization*, and *aggregation* to model relationships between security patterns, for example, a *Firewall* pattern [11] is the generalization of an *Address Filter Firewall* [20] and an *Application Firewall* [21] pattern.

They also propose two instances for the meta-model which represents two points of view, namely pattern-to-pattern relations, represented as a pattern graph, and pattern-to-dimension relations modelled as a dimension graph. They tested their approach with only eight different security patterns, which are close to implementation patterns.

To summarize, the published classifications take only a small number of security patterns into account. The used security patterns are often very similar to the first introduced patterns by Yoder and Barcalow [3]. Thus they do not consider

 ISBN: 978-1-61208-158-8

other types of security patterns such as enterprise patterns and mostly imply that only programming issues are covered by security patterns. Due to the fact that many people, who are involved in the software life-cycle, are rarely security experts and not aware of security issues [22], the classification should be easy to use. Based on these points, we will form a new classification for this purpose.

IV. New Classification for Security Patterns

Our goal is to present a classification that covers the heterogeneity of the published security patterns till today. It unifies the existing patterns in a common scheme. In addition, not every task needs information about attack surfaces or vulnerability classification properties like STRIDE or other facets that are introduced in Section III. On that account we omit specialized criteria like STRIDE and focus on obvious differences among the security patterns. With this in mind, we develop a new classification with a more general perspective based on a domain criterion (see Section II) and the security patterns we collected in our systematic literature review.

A. Systematic Literature Review

This section describes how the literature survey was conducted systematically [23], [24]. We start our literature research with the surveys by Laverdiere et al. [25], Heyman et al. [26] and Yoshioka et al. [27]. Additionally, we considered common pattern-related conferences and found 1150 articles (see Table I). In addition, we looked for security patterns in the IEEE Digital Library [28] and the ACM Digital Library [29] and took two books about security patterns [11] [14] into account.

We skimmed the conferences and electronic publications of the years 1997 to 2010 for several keywords such as cryptographic, security, software or secure. At first we picked out all publications that contain these keywords, secondly we read the abstract if it described the presentation of a security pattern. Finally, we read the publications, which are not filtered out previously to verify that they describe security patterns. Moreover, we scanned and collected the referenced publication of each identified pattern for further readings.

In summary, we identified 63 different publications describing security patterns, including books, journals, proceedings, and technical reports. Most of them were found by looking at the Hillside Group [30] pattern conferences such as PLoP and EuroPLoP (see Figure 2 and Table I) and books. Another publication type containing many security patterns were technical reports discovered by cross-references. New conferences that have only few security pattern publications are also discovered by cross-references such as International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), Working Conference on Data and Applications Security (IFIP WC 11.3), International Conference on Internet Computing (ICOMP), and International Workshop on Security, Trust and Privacy in Grid Systems (GRID-STP). The search at the ACM and IEEE Digital Library produced many false-positive articles that were at a closer look

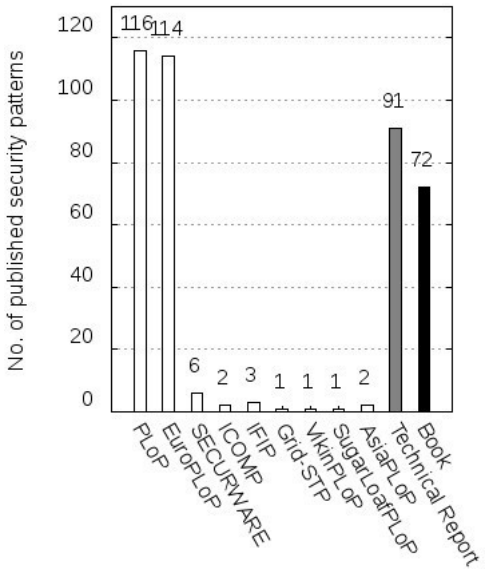


Fig. 2. Distribution of security patterns across different venues; white bars denote conferences, the grey bar technical reports, and the black bar books.

no security pattern descriptions but deal with them in other ways like discussing secure software design in practice [31].

Some publications describe more than one pattern. In total, we got 409 security patterns. This list depicted that some of these patterns have been described twice. Hence, we filtered out duplicates and reduced the number of patterns to 360. These duplicates were identified by the use of similar names and then comparing their descriptions. Because of the abundance of patterns, we were not able to check in depth whether two patterns with different names relate to the same concept. Based on this heterogeneous security pattern collection we formed our classification bottom-up in contrast to the aforementioned classifications in Section III.

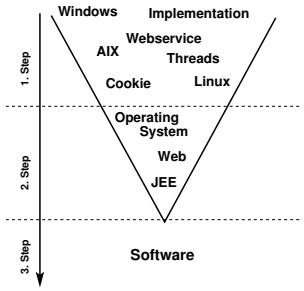


Fig. 3. Proceeding steps in our classification modelling.

B. Organizing by Application Domain

First of all, we skimmed over the data and collected keywords for the security patterns such as user, password, operating system, enterprise or process. These keywords were inspired by information we found in the pattern descriptions.

In the next iteration, we went through the pattern list and extracted keywords for the patterns. On further reading these

Acronym	Description	Total no. of articles 1997 to 2010	Articles describing security patterns
PLoP	Conference on Pattern Languages of Programs	427	26
EuroPLoP	European Conference on Pattern Languages of Programs	395	18
VikingPlop	The Nordic Conference on Pattern Languages of Programs	57	1
KoalaPLoP	Australian Conference on Pattern Languages of Programs	17	0
AsianPLoP	Asian Conference on Pattern Languages of Programs	14	1
SugarLoafPLoP	American Conference on Pattern Languages of Programming	156	1
PATTERNS	International Conferences on Pervasive Patterns and Applications	14	0

TABLE I
CONFERENCES INVOLVED IN THE INITIAL ARTICLE SELECTION.

keywords were unified into common groups. For instance, we united the keywords *AIX*, *Linux* and *Preforking* to the group *Operating System*. The result contains a mixture of purpose and domain criterion. We formed 13 different groups this way. To further simplify the classification along the lines like described in Section II, these keywords were further condensed to form an application-domain based distinction, which is easy to understand and intuitively applicable (see Figure 3). Finally, Figure 4 depicts the five target application domains which were discovered: *Enterprise*, *Software*, *Cryptographic*, *User*, and *Network*. They are described in the following in more detail.

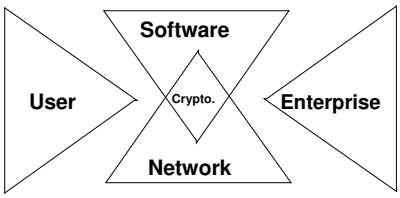


Fig. 4. Application-domain based classification.

Enterprise security patterns deal with aspects that are important for enterprises to ensure security in several enterprise segments like third party communication with suppliers. This means security in processes, physical authentication to several areas, risk mining or securing communication in inter- and external businesses. A good example of this pattern type is the *Manage Risk* pattern introduced by Elsinga and Hofman [32]. The problem addressed by this pattern is as follows "What is the right (combination of) paradigm(s) to formulate the corporate security strategy in order to select and implement the appropriate set of security safeguards?" The pattern suggests to instruct people and units to pay attention on known and unknown risks to develop prevention and roll-back strategies.

Network security patterns picture network infrastructures and their ideal composition, for example, using a *Packet Filter Firewall* to shield an internal network from Internet attacks or just tunnelling the communication traffic though a single controllable instance [11].

User security patterns are focused on user behaviour or awareness of security issues, for example, the *password lock box* pattern, which encourages the user to protect master passwords with the highest level of security [38]. It stresses the significance of protecting master password files and depict situations where such a file can be useful.

Software security patterns describe mostly how to structure parts of software to ensure security requirements. Sometimes they also describe a specific behaviour or way to manage or control a data flow in a secure way. On one hand, patterns in this domain can be very specific like JEE patterns, which can be applied only at Java enterprise applications [14]. On the other hand, patterns in this domain can be more general like the *Single Access Point* pattern, which models a kind of login structure that can be be found in several software systems like UNIX, ICQ or Twitter [3].

Cryptographic security patterns depict secure communication between two applications over a network. They are often described abstractly. Therefore, it is not clear whether these patterns reside in the *Network* or *Software* domain. Their implementation or application is possible in both domains. On that account, we see them as a part of network and software in our classification (see Figure 4). An example is the *Sender Authentication* pattern. It presents the problem and solution how to guarantee that a received message has been sent by a person one expected [41]. Obviously, such a pattern can be applied at network level (Level 3 and 4) or at application level and depending on that it resides on the *Network* or *Software* application domain.

The aforementioned classifications in Section III cover only

Application Domain	Publications Describing Security Patterns	Total no. of Security Patterns
Enterprise	[33], [34], [32], [35], [36], [37], [11]	84
User	[38], [39], [40]	23
Cryptographic	[41], [42], [43], [44]	35
Network	[45], [46], [21], [47], [48], [49], [50], [51], [52], [20], [53], [54], [55], [56], [57], [11], [58], [59], [60]	56
Software	[45], [50], [61], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [77], [78], [37], [79], [40], [11], [80], [14], [81], [3], [82], [62], [83]	162

TABLE II
PUBLICATIONS AND NUMBER OF SECURITY PATTERNS PER APPLICATION-DOMAIN

parts of the fields we discovered. The *Network* domain is partly touched by the classification of Konrad et al. [10]. Schumacher et al. factor *Enterprise* requirements customizable with view points in their classification, but they do not distinguish other domains as our approach does. The domains *User* and *Cryptographic* are not mentioned in the existing classification approaches, although they represent approximately one sixth of the patterns.

Our classification scheme can be tailored further to practical or research interests by employing view points as recommended by Fernandez et al. [18]. For software engineering in particular, applicable patterns are located in the category *Software*, which can be further divided into specific purposes such as pattern detection by using the existing pattern classifications by Gamma et al. [2] or Shi and Olsson [84]. Developing new view points or finer grained classifications to cover new needs in terms of special purposes for one of the application domains is also conceivable.

V. CONCLUSION AND OUTLOOK

In this paper, we presented our systematically literature review on security patterns and introduced a new classification scheme. It embraces 360 published security patterns and exceeds existing classifications by far. The presented classification scheme fulfils the requirements of classifications in the terms of expandability, intuitive use, and is applicable for security laymen.

We hope that this classification will support our ongoing research toward security patterns and reverse engineering. In particular, we plan to investigate software architectures in terms of security pattern usage. Further, we expect that this classification helps other researchers and practitioners with specific application goals focussing on security patterns.

VI. ACKNOWLEDGMENTS

This work was supported by the German Federal Ministry of Education and Research (BMBF) under the grant 01IS10015B (ASKS project).

REFERENCES

[1] The H Security, "Number of critical, but unpatched, vulnerabilities is rising," 2010, (28.08.2011). [Online]. Available: http://www.h-online.com/security/news/item/Number-of-critical-but-unpatched-vulnerabilities-is-rising-1067495.html
[2] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Object-Oriented Software*. Addison Wesley, 1994.
[3] J. Yoder and J. Barcalow, "Architectural patterns for enabling application security," in *Proc. of PLOP*, 1997.
[4] S. Haldikis, A. Chatzigeorigou, and G. Stephanides, "A practical evaluation of security patterns," in *Proceedings of AIDC*, 2006.
[5] M. Hafiz and R. E. Johnson, "Evolution of the mta architecture: the impact of security," *Softw. Pract. Exper.*, vol. 38, no. 15, 2008.
[6] M. Hafiz, P. Adamczyk, and R. E. Johnson, "Organizing security patterns," *IEEE Software*, vol. 24, 2007.
[7] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture: A System of Patterns*. Wiley, 1996.
[8] E. B. Fernández, N. Yoshioka, H. Washizaki, and J. Juerjens, "Using security patterns to build secure systems," in *Proc. of 1st SPAQu*, 2007.
[9] D. M. Kienzle and M. C. Elder, "Final technical report: Security pattern for web application development," Tech. Rep., 2002, 27.04.2011. [Online]. Available: http://www.scrypt.net/~celer/securitypatterns/final%20report.pdf
[10] S. Konrad, B. Cheng, L. Campbell, and R. Wassermann, "Using security patterns to model and analyze security requirements," in *Proc. of RHAS*, 2003.
[11] M. Schumacher, E. B. Fernandez, D. Hybertson, and F. Buschmann, *Security Patterns: Integrating Security and Systems Engineering*. John Wiley & Sons, 2005.
[12] "The zachmann framework for enterprise architecture," 2011, (28.08.2011). [Online]. Available: http://zachmaninternational.com/2/Zachman_Framework.asp
[13] D. Alur, D. Malks, and J. Crupi, *Core J2EE Patterns: Best Practices and Design Strategies*. Prentice Hall PTR, 2001.
[14] C. Steel, R. Nagappan, and R. Lai, *Core Security Patterns: Best Practices and Strategies for J2EE(TM), Web Services, and Identity Management*. Prentice Hall International, 2005.
[15] D. G. Rosado, C. Gutiérrez, E. Fernández-Medina, and M. Piattini, "Security patterns related to security requirements," in *Proc. of WOSIS*, 2006, pp. 163–173.
[16] F. Swiderski and W. Snyder, *Threat Modeling (Microsoft Professional)*. Microsoft Press, 2004.
[17] M. VanHilst, E. B. Fernandez, and F. A. Braz, "A multi-dimensional classification for users of security patterns," in *Proc. of WOSIS*, 2008, pp. 89–98.
[18] E. B. Fernandez, H. Washizaki, N. Yoshioka, A. Kubo, and Y. Fukazawa, "Classifying security patterns," in *Proc. of Asian-Pacific Web Conference*, Apr. 2008, pp. 342–347.

 ISBN: 978-1-61208-158-8

[19] H. Washizaki, E. B. Fernandez, K. Maruyama, A. Kubo, and N. Yoshioka, "Improving the classification of security patterns," in *Proc. of DEXA*. Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 165–170.

[20] E. B. Fernandez, M. M. Larrondo-petrie, N. Seliya, N. Delessy, and A. Herzberg, "A pattern language for firewalls," in *Proc. of PLoP*, September 2003.

[21] S. R. Nelly Delessy-Gassant, Eduardo B. Fernandez and M. M. Larrondo-Petrie, "Patterns for application firewalls," in *Proc. of PLoP*, 2004.

[22] K. van Wyk and G. McGraw, "Bridging the gap between software development and information security," *Security Privacy, IEEE*, 2005.

[23] B. Kitchenham, "Procedures for performing systematic reviews," Keele University, Keele, UK, Technical Report TR/SE-0401, 2004.

[24] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele University, Keele, UK, Technical Report EBSE-2007-001, 2007.

[25] M. Laverdiere, A. Mourad, A. Hanna, and M. Debbabi, "Security design patterns: Survey and evaluation," in *Procs. of CCECE*, 2006.

[26] T. Heyman, K. Yskout, R. Scandariato, and W. Joosen, "An analysis of the security patterns landscape," in *Proc. 3rd SESS*. Washington, DC, USA: IEEE Computer Society, 2007, p. 3.

[27] N. Yoshioka, H. Washizaki, and K. Maruyma, "A survey on security patterns," *Progress in Informatics*, vol. 5, pp. 35–47, 2008.

[28] IEEE Digital Library, 2011, (28.08.2011). [Online]. Available: http://www.computer.org/portal/

[29] ACM Digital Library, 2011, (28.08.2011). [Online]. Available: http://portal.acm.org/

[30] The Hillside Group, 2011, (28.08.2011). [Online]. Available: http://hillside.net

[31] P. Meland and J. Jensen, "Secure software design in practice," in *Proc. of ARES*, march 2008, pp. 1164 –1171.

[32] B. Elsinga and A. Hofman, "Security paradigm pattern language," in *Proc. of EuroPLoP*, 2003.

[33] G. Dallons, P. Massonet, J. f. Molderez, C. Ponsard, and A. Arenas, "An analysis of the chinese wall pattern for guaranteeing confidentiality in grid-based virtual organisations," in *Proc. of Grid-STP*. IEEE, 2007.

[34] P. Dyson and A. Longshaw, "Patterns for managing internet-technology systems," in *Proc. of EuroPLoP*, 2003.

[35] A. M. Ernst, "Enterprise architecture management patterns," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008, pp. 1–20.

[36] E. B. Fernandez, J. Ballesteros, A. C. Desouza-Doucet, and M. M. Larrondo-Petrie, "Security patterns for physical access control systems," in *Proc. of IFIP WG 11.3*. Springer-Verlag, 2007, pp. 259–274.

[37] S. Romanosky, "Security design patterns part 1," in *Proc. of PLoP*, 2001.

[38] D. Riehle, W. Cunningham, J. Bergin, N. Kerth, and S. Metsker, "Password patterns," in *Proc. of EuroPLoP*, 2002.

[39] S. Romanosky, A. Acquisti, J. Hong, L. F. Cranor, and B. Friedman, "Privacy patterns for online interactions," in *Proc. of PLoP*. New York, NY, USA: ACM, 2006.

[40] M. Schumacher, "Security patterns and security standards - with selected security patterns for anonymity and privacy," in *Proc. of EuroPLoP*, 2002.

[41] A. Braga, C. Rubira, and R. Dahab, "Tropyc: A pattern language for cryptographic software," in *Proc. of PLoP*, 1998.

[42] A. Cuevas, P. E. Khoury, L. Gomez, and A. Laube, "Security patterns for capturing encryption-based access control to sensor data," *Proc. of SECURWARE*, pp. 62–67, 2008.

[43] Sami Lehtonen and Juha Pärssinen, "A pattern language for cryptographic key management," in *Proc. of EuroPLoP*, 2002.

[44] S. Lehtonen and J. Pärssinen, "A pattern language for key management," in *Proc. of PLoP*, 2001.

[45] B. Blakley and C. Heath, *Security Design Patterns*. The Open Group, Apr. 2004, 27.04.2011. [Online]. Available: www.opengroup.org/onlinepubs/9299969899/toc.pdf

[46] A. Cuevas, P. El Khoury, L. Gomez, A. Laube, and A. Sorniotti, "A security pattern for untraceable secret handshakes," in *Proc. of SECURWARE*, Jun. 2009, pp. 8 –14.

[47] N. Delessy and E. B. Fernandez, "Patterns for the extensible access control markup language," in *Proc. of PLoP*, 2005.

[48] M. Schumacher, "Firewall patterns," in *Proc. of EuroPLoP*, 2003.

[49] E. B. Fernandez, J. C. Pelaez, and M. M. Larrondo-Petrie, "Security patterns for voice over ip networks," in *Proc. of ICCGI*. IEEE Computer Society, 2007.

[50] N. Delessy, E. B. Fernandez, M. M.Larrondo-Petrie, and J. Wu, "Patterns for access control in distributed systems," in *Proc. of PLoP*, 2007.

[51] L. B. Jr, F. L. Brown, J. Divietri, G. D. D. Villegas, and E. B. Fernandez, "The authenticator pattern," in *Proc. of PLoP*, 1999, p. 6.

[52] E. B. Fernandez and R. Warrier, "Remote authenticator / authorizer," in *Proc of. PLoP*, 2003.

[53] M. Hafiz, "A collection of privacy design patterns," in *Proc. of PLoP*. New York, NY, USA: ACM, 2006, pp. 1–13.

[54] M. Schumacher and U. Roedig, "Security engineering with patterns," in *Lecture Notes in Computer Science*. Springer, 2001.

[55] T. Okubo and H. Tanaka, "Web security patterns for analysis and design," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008.

[56] M. Sadicoff, M. M. Larrondo-petrie, and E. B. Fernandez, "Privacy-aware network client pattern," in *Proc. of PLoP*, 2005.

[57] B. Schleinzer and N. Yoshioka, "Security pattern for data integrity in p2p systems," in *Proc. of PLoP*, 2010.

[58] P. Sommerlad, "Reverse proxy patterns," in *Proc. of EuroPLoP*, 2003.

[59] S. Romanosky, "Enterprise security patterns," in *Proc. of EuroPLoP*, 2002.

[60] A. Kumar and E. Fernandez, "A security pattern for a virtual private network," in *Proc. of SugarLoafPLoP*, 2010.

[61] C. Dougherty, K. Sayre, R. C. Seacord, D. Svoboda, and K. Togashi, "Secure design patterns," Carnegie Mellon University, Software Engineering Institute, report 2009-TR-010, Oct. 2009.

[62] B. Elsinga and A. Hofman, "Control the actor-based access rights," in *Proc. of EuroPLoP*, 2002.

[63] E. B. Fernandez and J. Sinibaldi, "More patterns for operating systems access control," in *Proc. of EuroPLoP*, 2003.

[64] E. B. Fernandez and T. Sorgente, "A pattern language for security models," in *Proc. of PLoP*, 2001.

[65] E. B. Fernandez, "Patterns for operating systems access control," in *Procs. of PLoP*, 2002.

[66] E. B. Fernandez, T. Sorgente, and M. M. Larrondo-Petrie, "Even more patterns for secure operating systems," in *Proc. of PLoP*. ACM, 2006.

[67] E. B. Fernandez and D. laRed Martinez, "Patterns for the secure and reliable execution of processes," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008, pp. 1–16.

[68] E. B. Fernandez and G. Pernul, "Patterns for session-based access control," in *Proc. of PLoP*. ACM, 2006.

[69] V. Gondi, "Multiple secure observers using j2ee," in *Proc. of PLoP*, 2010.

[70] M. Hafiz, "Secure pre-forking - a pattern for performance and security," in *Proc. of PLoP*, 2005.

[71] M. Hafiz, R. E. Johnson, and R. Af, "The security architecture of qmail," in *Proc. of PLoP*, 2004.

[72] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt, "Security patterns repository, version 1.0," 2003, (28.08.2011). [Online]. Available: http://www.scrypt.net/~celer/securitypatterns/repository.pdf

[73] S. R. Kodituwakku, P. Bertok, and L. Zhao, "Aplrac: A pattern language for designing and implementing role-based access control," in *Proc. of EuroPLoP*, 2001.

[74] Q. H. Mahmoud, "Security policy: A design pattern for mobile java code," in *Proc. of PLoP*, 2000.

[75] H. Mouratidis, P. Giorgini, and M. Schumacher, "Security patterns for agent systems," in *Proc. of EuroPLoP*, 2003.

[76] P. Morrison and E. B. Fernandez, "The credentials pattern," in *Proc. of PLoP*. New York, NY, USA: ACM, 2006, pp. 1–4.

[77] Patrick Morrison and Eduardo B. Fernandez, "Securing the broker pattern," in *Proc. of EuroPLoP*, 2006.

[78] J. L. Ortega-Arjona and E. B. Fernandez, "The secure blackboard pattern," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008.

[79] T. Saridakis, "Design patterns for fault containment," in *Proc. of EuroPLoP*, 2003.

[80] K. E. Sørensen, "Session patterns," in *Proc. of EuroPLoP*, 2002.

[81] M. Weiss, "Credential delegation: Towards grid security patterns," in *Proc. of The Nordic Conference on Pattern Languages of Programs*, 2006.

[82] Y. Zhou, Q. Zhao, and M. Perry, "Policy enforcement pattern," in *Procs. of PLoP*, 2002.

[83] E. B. Fernandez, S. Mujica, and F. Valenzuela, "Two security patterns," in *Proc. of Asian Conference on Pattern Languages of Programs*, 2010.

[84] N. Shi and R. A. Olsson, "Reverse engineering of design patterns from java source code," in *Procs. of 21st ASE*. IEEE Computer Society, 2006.

 ISBN: 978-1-61208-158-8

Patterns Combining Reliability and Security

Ingrid A. Buckley, Eduardo B. Fernandez, and Maria M. Larrondo-Petrie
Dept. of Computer & Electrical Engineering and Computer Science
Florida Atlantic University
Boca Raton, USA
ibuckley@fau.edu, ed@cse.fau.edu, petrie@fau.edu

***Abstract*—We are developing a methodology to combine security and reliability. One aspect of this fusion implies developing patterns that combine both objectives. The Secure Reliability (SecRel) and Reliable Security (RelSec) are hybrid patterns that combine security and reliability. The SecRel pattern applies security to control the functions of a reliable system, while the RelSec pattern applies reliability to the functions that provide security. We show how these patterns relate to our methodology, and in which architectural levels they could be used.**

Keywords-*software lifecycle; software patterns; reliability; reliability patterns; security; security patterns.*

I. INTRODUCTION

Reliability is a key system characteristic that is an increasing concern for current systems. Greater reliability is necessary due to the new ways in which services are delivered to the public. Services are used by many industries, including health care, government, telecommunications, tools, and products. The lack of reliability in many systems has encouraged research efforts to find ways to improve this situation. Applications have become very complex and their reliability is a current concern.

Typically, reliability is provided through redundancy, checking and monitoring, aspects which are usually added after a system is built. A good amount of work, e.g. [11, 12, 14], has been done to include reliability in systems. There is also a large amount of work on security patterns [17]. Similarly as we did for security [8], we propose here adding reliability throughout the software development life cycle. In our approach, we start by identifying the possible failures in a system. By analyzing UML activity diagrams for all use cases and considering possible failures in each activity, we can enumerate possible service failures in applications [5]. Once failures are identified, we apply appropriate policies, realized as patterns, which will stop or mitigate these failures. In some critical parts of a system we also want to be able to provide security and reliability at the same time.

We can combine security and reliability using patterns. A pattern is an encapsulated solution to a recurrent problem in a given context. Design patterns [10] embody the experience and knowledge of many designers and when properly catalogued, they provide a repository of solutions for useful problems. Initially used for improving code, patterns are becoming more and more used to build secure and reliable systems [3, 6]. We present here two of these patterns. The Secure Reliability (SecRel) pattern applies security to a reliable system, while the Reliable Security (RelSec) pattern applies reliability to the functions that provide security in a secure system.

Section 2 discusses an approach for a secure and reliable lifecycle considering space and time aspects, including a metamodel for reliability requirements, Sections 3 and 4 present the RelSec and SecRel patterns, respectively. Section 5 provides some conclusions.

II. RELIABILITY IN SPACE AND TIME

A good way to define precise relationships between concepts in software development is to express them in metamodels. Our approach involves enumeration of failures and their origins and finds policies and patterns to handle them, so we can express their relationship as shown in Figure 1. A **Fault** manifests itself as an **Error**. If the error is not contained, it can manifest itself into a **Failure,** which indicates that the system is not following its specifications [1]. A **Policy** can avoid or handle a failure. A **Pattern** realizes the policy that can handle the fault; some examples of reliability patterns are given in [3, 4]. If we can enumerate all faults and have appropriate patterns to handle them, we can build reliable systems. We can define also patterns needed to comply with regulations.

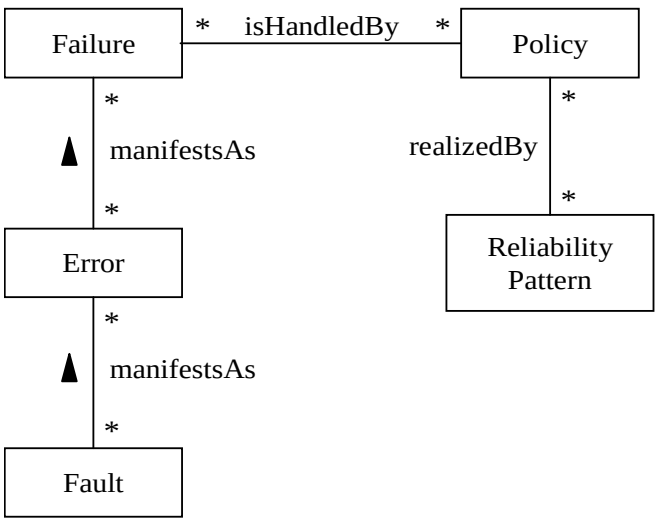


Figure 1. A metamodel for reliability concepts

 ISBN: 978-1-61208-158-8

The development of software applications starts from the requirements expressed normally as use cases. The use cases can be converted into a conceptual or analysis model. Analysis is a fundamental stage since the conceptual model can be shown to satisfy the requirements and becomes the skeleton on which the entire system can be built.

Reliability aspects in a computer system are expressed differently at different architectural layers (levels) and stages of the software development process. Figure 2 shows where we need to apply reliability along the stages of the system lifecycle and the architectural levels. An appropriate degree of reliability is required in each tier of the computer system layers and in each stage; also the expression of reliability varies. An important point illustrated in this diagram is the fact that the requirements must be carried over along time and along space:

1. User Interface – This is the highest level and is the user's point of contact for with the system. Usability is an important aspect for reliability and should be reflected in the interfaces.
2. Applications – The application tier of the system consists of services and programs that carry out useful operations. This tier invokes functions that are requested by the user, and should always be available when needed.
3. Middleware – Manages the interactions between the applications, DBMS, OS, and users of a system.
4. Database Management System (DBMS) – this tier is where data is stored. Failures may affect most of the applications.
5. Operating System (OS) – This level manages and synchronizes all the functions and resources within the system. Its reliability is fundamental because failures here affect all the applications
6. Data Communication – This is part of the OS and manages how information is passed throughout the system.
7. Hardware - This is the lowest level of the architecture, where instructions are executed. Its failures affect the whole system.

The requirements define the degree of reliability that the application needs. In the analysis stage, given the reliability requirements, we match the identified failures to the set of reliability mechanisms needed to stop these failures identified. If the failure is hardware based then redundancy and diversity are appropriate. If the failure is based on software, we need diversity or other approaches.

The design stage is governed by the reliability mechanisms identified in the analysis stage, which are selected to prevent failures. This may have two or more levels to describe implementation-oriented features. Web services and clouds are typical distributed architectures used in practice.

The implementation stage follows directly after the design stage; here reliability is realized in the form of code and COTS components. The whole process is iterative where some stages or parts may need to be redone. Our idea is to combine security and reliability aspects in each stage and each level. We realize reliability mechanisms by applying patterns and a catalog of them is needed to assist the architects and designers when building the system. We have produced some reliability patterns [3, 4]; here we add two patterns that combine reliability and security.

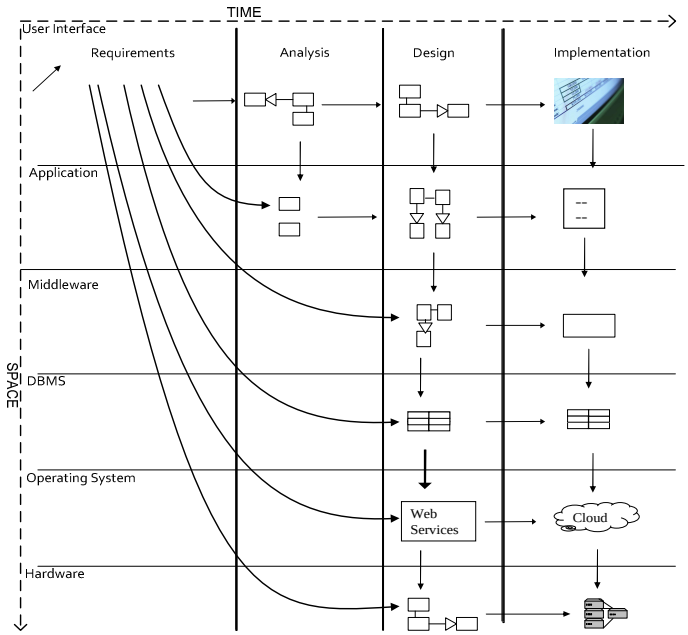


Figure 2. Reliability in Space and Time

III. THE SECURE RELIABILITY PATTERN

Intent

This pattern intends to control the use of reliable services or services that have a direct impact on system reliability. We can have services implemented using different reliability mechanisms but users may have access to only some of them. The misuse of some services may have a strong effect on system reliability.

Example

Consider a SCADA system which consists of field units, a central controller and communications networks. The field units are controlled by the central controller which is usually connected to corporate networks and or the Internet. Attackers may be able to input commands to the field units resulting in damage or disruption.

 ISBN: 978-1-61208-158-8

Context

Critical systems or applications that need a high degree of security and reliability to operate successfully and where we have services implemented with different degrees of reliability according to their criticality.

Problem

We can have services implemented using different reliability mechanisms but users may have access to only some of them. The misuse of some services may have a strong effect on system reliability. How do we restrict the use of the system services that affect reliability? The solution to this problem is affected by the following forces:

Reliability:

- We need to control the level of reliability of the system.
- We can have different implementations of some services according to their criticality.
- Some services affect the system reliability if improperly used; that is, security attacks can affect the reliability of the system.
- The total overhead should be reasonable.

Security:

- The system requires a given level of security. Errors can affect the security of the system.

Solution

Separate those services which could affect the reliability of the system and apply to them Role-Based Access Control [16] so that only authorized roles can use them; apply also a least-privilege policy [6]. This protects the confidentiality and integrity of the service. The structure of this pattern is illustrated in Figure 1.

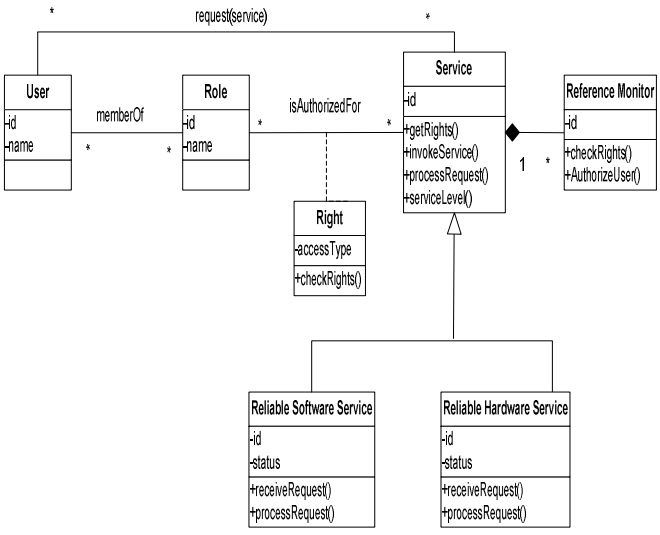


Figure 3. Class Diagram for the Secure Reliability Pattern

Structure

In Figure 3, every user is a member of a **Role**, and each Role has specific **Rights** associated with it. The **Service** entity implements a Strategy pattern [10], which chooses a **Software** or **Hardware Service,** depending on the needs of the application. The **Reference Monitor** enforces the authorized use of the service.

Dynamics

One of the dynamic aspects of the SecRel pattern is described using a sequence diagram for the use case "User sends a request for a service", shown in Figure 4.

UC: User sends a request for a service.
Summary: A user requests a reliability-sensitive service. The request is validated by a Reference Monitor.
Actors: User
Description:

1. A user requests a service with a given level of reliability or reliability-sensitive.
2. The service invokes a Reference Monitor to check if the user is authorized.
3. If the user is authorized, the request is passed to the service. If not authorized, the request is rejected with a violation message to the user.
4. The requested service processes the request.

Post condition:
The request has been processed or rejected.

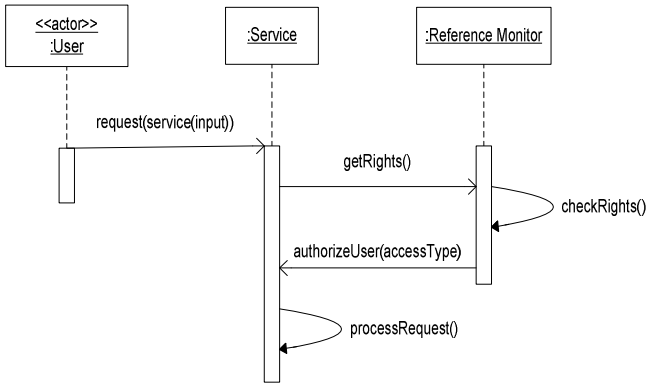


Figure 4. Sequence Diagram for the UC "User requests a service"

Implementation

To implement the Secure Reliability pattern the following is required:

1. We must have a set of reliable mechanisms or a set of reliability-sensitive services.
2. A reference monitor system is required to check the rights associated with each role before a service is used.
3. The reference monitor or an authorization service must perform authentication before a user is authorized

 ISBN: 978-1-61208-158-8

4. If the user is remote, a secure channel can be employed to ensure that the request of a user is passed securely so as to protect the request in transit.

Consequences

The SecRel pattern presents the following advantages:

- We can control the confidentiality and integrity of reliability-sensitive services.
- We can control the use of different degrees of service reliability by selecting different reliability mechanisms.
- The overhead of access control is small.

The pattern also has some possible liabilities:

- The authentication and authorization of users take time.
- Services may be replicated to increase their reliability; however this can increase the system overhead in maintaining them.

Known Uses

- Motorola's Canopy Platform is a wireless broadband system which enables extending broadband networks to deploy data, voice, and video applications [13]. This product uses part of the components used in the SecRelc pattern.
- Boeing's P-8 is a military derivative of the Boeing Next-Generation 737-800. It is an advanced anti-submarine and anti-surface warfare aircraft [2]. P-8 uses part of the SecRel pattern.

Related patterns

- Various reliability and fault tolerance patterns which include the Active Replication [2], TMR (Triple Modular Redundancy), and NVP (N-Version programming patterns.
- This pattern can be seen as a variation of the Role-Based Access Control pattern of [17].
- We need Authentication before we can apply Authorization [17].

IV. THE RELIABLE SECURITY PATTERN

Intent

This pattern intends to perform reliable authorization enforcement by applying reliability mechanisms to the Reference Monitor and to the Authorization rules.

Example

Consider a SCADA system which consists of field units, a central controller and communications networks. The field units are controlled by the central controller which is usually connected to corporate networks and or the Internet. Because operations can be carried out over a network, this raises a security concern. Also, the field units are usually in separate geographical locations from the central controller, therefore extreme weather or tampering can affect them. Usually it is also difficult to access them physically to repair damages, which raises a general reliability concern. If an error occurs in the security system, attackers can perform malicious actions that can disrupt the operation of the system.

Context

Critical systems or applications that need a high degree of security and reliability to operate successfully and where we have services implemented with different degrees of reliability according to their criticality.

Problem

How can we ensure that authorization is always performed correctly in the presence of errors? The solution to this problem is affected by the following forces:

Security:

- The system should always perform authorization correctly in the presence of errors. Otherwise, we will have security violations.
- The total overhead of the reliability mechanisms should be reasonable.

Reliability:

- Security services should define and enforce security constraints in a reliable way.

Solution

Apply reliability mechanisms to the Reference Monitor and to the Authorization rules.

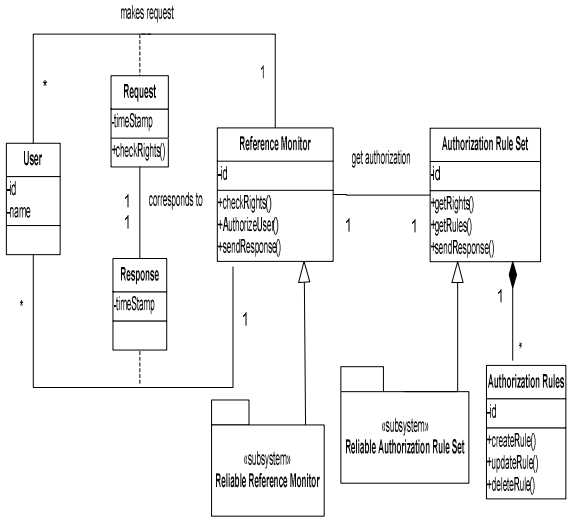


Figure 5. Class Diagram for the Reliable Security Pattern

Structure

The structure of this pattern is illustrated in Figure 5. All user **Requests** are evaluated by the **Reference Monitor,** which has

access to the **Authorization Rule Set**. When the user sends a request, the **Reference Monitor** intercepts it and checks the rights (rules) associated with the request.

The **Reliable Reference Monitor** incorporates standard reliability mechanisms [3], The **Authorization Rule Set** also includes a reliability mechanism. The **Response** defines the decision from the Reference Monitor and must match the Request.

Dynamics
Figure 6 shows a sequence diagram for the use case "User requests a service".

UC: User requests a service.
Summary: A user sends a request, and the request is validated by a Reference Monitor.
Actors: User
Description:
1. The user requests some service.
2. If the user is authorized his request is processed.
3. A response is sent in response to the user's request.

Post condition:
The request has been approved or rejected.

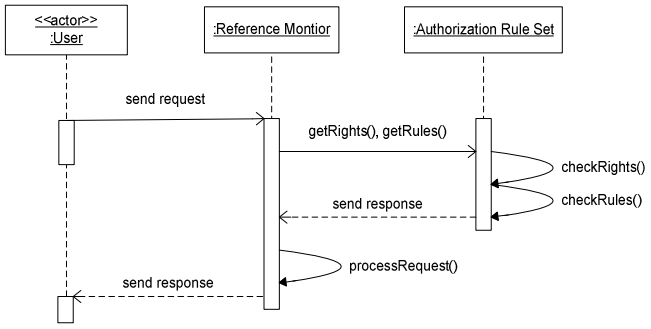


Figure 6. Sequence diagram for the UC "User requests a service"

Implementation

To implement the Reliable Security pattern the following is required:
1. We need a variety of reliability mechanisms that can be applied to the security services, e.g., those described in [3, 4].
2. The reference monitor must perform authentication before a user can send a request.
3. If the user is remote, a secure channel can be employed to ensure that the request of a user is passed securely so as to protect the request in transit.

Consequences

The RelSec pattern presents the following advantages:
- We can implement different degrees of reliability for the security services by selecting different mechanisms.
- Security checks will be reliable because all security components are reliable.
- The overhead of access control can be small because we can use reliability mechanisms that are not very complex and can be controlled to have small overhead.

The pattern also has some possible liabilities:
- The reliability mechanisms add some overhead.
- The system is more complex compared to a system without reliability mechanisms.

Known Uses

- Trumba Connect is web-hosted active event publishing solution that provides organizations with a two-way communication vehicle between events published on their websites and the personal calendaring systems used by their site visitors [19]. It uses part of the RelSec pattern.

Related patterns

- The Authentication pattern provides facilities for authenticating a user in a system [17].
- The Authorization pattern provides a way to define authorization for the users to the resources of a system [17].
- The Reference Monitor pattern checks if the process has the rights to access the object [6, 17].
- The Strategy pattern can be used to select the most suitable options to apply reliability [10].

V. DISCUSSION

The use of patterns is still not widespread in industrial institutions. Design patterns are used in large companies but many smaller companies only do coding, they don't even use models. Our work tries to encourage the use of models to build complex systems; building these systems without models will result in systems which are faulty and insecure. Models allow catching errors early in the lifecycle, which results in development savings [15]. Security patterns have reached a mature level and are starting to be used in industry for secure software development. Reliability patterns are less developed and no catalogs exist yet. Both types of patterns can help developers who are not security or reliability experts to build better systems by providing them with packaged proven solutions. Combining security and reliability in the form of patterns makes their work even easier for some applications where we need services with these two aspects. However, isolated patterns are not useful, we need pattern classification approaches and complete software development methodologies. The use of patterns fits well with Model-Driven approaches [14]; we can define reliable and secure services in domain models that can be used as starting points for critical applications [9].

 ISBN: 978-1-61208-158-8

VI. RELATED WORK

There is a significant amount of work on security patterns, including catalogs of patterns [17], surveys [21], and analysis of their uses and possibilities, e.g., [20]. The same is not true for reliability patterns; only a few patterns have appeared, e.g., [3, 16]. Security patterns have shown to be quite useful for designing secure systems [8] and the same can be expected for reliability patterns. Before that happens, we need a good and complete catalog of reliability patterns. As far as we know, nobody has tried to combine security and reliability patterns, although methodologies trying to combine these two aspects already exist, e.g., [14]. Some work tries to build directly reliable architectures by applying the patterns through tactics [11]. Another approach tries to insert the patterns in specific points in the architecture [18]. We believe that a systematic methodology, considering all the stages of the lifecycle is necessary. Reliability, similarly to security, must be incorporated in all the stages of the software lifecycle, adding these features in the code has shown to be ineffective. Patterns like the ones presented here combine aspects of security and reliability and can be used in systems that require a high level of security or reliability at least in some services.

VII. CONCLUSIONS

There are situations, mostly in critical systems, where the need to apply security to reliable systems and where the authorization systems should not fail. The two patterns presented here attempt to combine aspects of reliability and security to allow the implementation of systems that require very high levels of both features. The patterns are a part of our methodology to build critical systems but also have independent value.

Patterns provide a clear way for inexperienced designers to add reliability or security into their designs, but they require a good catalog of patterns that can fit all the system needs; these two patterns can be part of such a catalog. We already have a fairly complete catalog of security patterns [7, 16], so we need to find more reliability patterns as well as combined patterns as those shown here. The patterns presented here can be used in the analysis stage but they need to be extended to reflect the environment where they will be used; e.g., reliability in web services [4]. These patterns also are relevant to any level of the architecture, implemented in the appropriate technology; for example, if we determine in the use cases if a particular service is highly critical, we can add one of these patterns to the conceptual model of the application; the reliable model can then be reflected to the database or operating system levels. If web services are used for distribution, the patterns can define reliable or secure architectures for some web services.

ACKNOWLEDGMENTS

The referees provided valuable suggestions that helped improve this paper.

REFERENCES

[1] A. Avizienis, J. C. Laprie, and B. Randell, “Fundamental concepts of dependability”, UCLA CSD Report No. 010028.

[2] Boeing, “P8”, http://www.boeing.com/defense-space/military/p8/index.html, 2010. (Last accessed: August 29, 2011)

[3] I. A. Buckley and E.B. Fernandez, "Three patterns for fault tolerance" , Procs. of the OOPSLA MiniPLoP, October 26, 2009. http://www.refactory.com/miniploppapers/FTPatts.pdf. (Last accessed: August 29, 2011)

[4] I. A. Buckley, E.B. Fernandez, G. Rossi, and M. Sadjadi, "Web services reliability patterns", short paper in the 21st International Conference on Software Engineering and Knowledge Engineering (SEKE'2009), Boston, July 1-3, 2009, pp. 4-9.

[5] I. A. Buckley and E.B. Fernandez, “Enumerating failures to build reliable systems”, submitted for publication.

[6] E. B. Fernandez, S. Mujica, and F. Valenzuela, "Two security patterns: Least Privilege and Secure Logger/Auditor.", Procs. of Asian PLoP 2011. http://patterns-wg.fuka.info.waseda.ac.jp/asianplop/program.html#papers. (last accessed August 30, 2011)

[7] E. B. Fernandez, “Patterns for operating systems access control”, Procs. of PLoP 2002, http://www.hillside.net/plop/plop2002/. (Last accessed: August 29, 2011)

[8] E. B. Fernandez, M.M. Larrondo-Petrie, T. Sorgente, and M. VanHilst, "A methodology to develop secure systems using patterns ", Chapter 5 in "Integrating security and software engineering: Advances and future vision", H. Mouratidis and P. Giorgini (Eds.), IDEA Press, 2006, pp. 107-126.

[9] E. B. Fernandez and S. Mujica, “Model-based development of security requirements”, accepted for the CLEI (Latin-American Center for Informatics Studies) Journal.

[10] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. Design patterns: elements of reusable object-oriented software, Boston, Mass:Addison-Wesley, 1994.

[11] N. Harrison, P. Avgeriou, and U. Zdun, “On the impact of fault tolerance tactics on architecture”, ACM SERENE 2010, London, UK, April 13-16, 2010, pp. 9-18, doi: 10.1145/1479772.1479775.

[12] M. R. Lyu, “An integrated approach to achieving high software reliability”, Procs. IEEE Aerospace Conference, Aspen, Colorado, March 21-28, 1998, vol. 4, pp. 123-136, doi: 10.1109/AERO.1998.682162.

[13] Motorola Inc, “Motorola’s Canopytm platform” http://www.ptsupply.com/pdf/motorola_canopy.pdf, 2004. (Last accessed: August 29, 2011)

[14] S. Mustafiz, X. Sun, and J. Kienzle, “Model-driven assessment of system dependability”, Software and Systems Modelling, vol. 7, 2008, pp. 487-502, doi: 10.1007/s10270-008-0084-1.

[15] OWASP, “OWASP Risk Rating Methodology”, https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology, 2010. (Last accessed: August 31, 2011)

[16] T. Saridakis, “A System of Patterns for Fault Tolerance”, Procs. of EuroPLoP, 2002, pp. 535–582.

[17] M. Schumacher, E. B. Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad, Security Patterns: Integrating security and systems engineering, Wiley Series on Software Design Patterns, Wiley, 2006.

[18] M. Tichy, “Pattern Based Synthesis of Fault Tolerant Embedded Systems”, Procs. of the Doctoral Symposium of the Fourteenth ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE), Portland, Oregon, USA, pp. 13-18. ACM Press, November 2006.

[19] Trumba Corporation, “What features does Trumba Connect offer?”, http://www.trumba.com/connect/ knowledgecenter/pdf/Features-list_SS-006.pdf, 2007.(Last accessed: August 29, 2011)

[20] R.Villarroel, E. Fernández-Medina, M. Piattini, “Secure information systems development-A survey and comparison”,

 ISBN: 978-1-61208-158-8

Computers & Security, vol. 24, No 4, pp. 308-321, doi: 10.1016/j.cose.2004.09.011.

[21] N. Yoshioka, H. Washizaki, and K. Maruyama, “A survey on security patterns”, Progress in Informatics, vol. 5, pp. 35-47, doi:10.2201/NiiPi.2008.5.5.

A Systematic Security Analysis of Information Systems

Roberto Ortiz, Santiago Moral-Rubio
Dep. Information Security
BBVA Group
Madrid, Spain
r.ortizpl@gmail.com;
santiago.moral@bbva.com

Javier Garzás
Kybele Group
Rey Juan Carlos University
Madrid, Spain
javier.garzas@urjc.es

Eduardo Fernandez-Medina
GSyA Research Group
University of Castilla- La Mancha
Ciudad Real, Spain
Eduardo.FdezMedina@uclm.es

***Abstract*—The integration of security into software development processes through methodologies guarantees that these developments are controlled, planned and verified at all stages. It is thus possible to avoid unexpected errors whilst improving the quality and security of the system produced. These methodologies can be enriched with the use of security patterns that compile the knowledge of security experts in a documented and structured manner, providing us with a systematic means to solve recurring problems. In this paper we shall summarize pattern-based security methodology in order to support both the construction of secure information systems and the maintenance of the level of the security attained, upon which we are currently working. We shall also provide an in–depth study of the analysis stage, showing the elements of which it is composed, such as the input and output artifacts, together with the main roles and activities that participate in it.**

Keywords-Security; Security Methodology; Secure Systems Analysis; Security Patterns; Secure Information Systems

I. INTRODUCTION

In recent years, the majority of attacks against organizations aim to exploit the vulnerabilities caused by a poor design and development of the functionalities given to information systems (IS) [1]. The need to build secure IS has therefore arisen, and this situation has encouraged the scientific community to research the integration of security into IS development processes [2-4]. This integration is established through development methodologies since they offer, from the first stages, a systematic, planned, controlled, verifiable and detailed process that will avoid the appearance of uncontrolled security errors, thus mitigating the possible risks associated with the implementation of new functionalities in an IS [5]. The main advantage of these processes is also based on the fact that they are decomposed into elementary tasks in which each task is identified by a procedure that defines how to carry it out, the most appropriate actors for its implementation and the tools and techniques needed in each one of them [6].

Given that the purpose of security methodologies consists of systematizing the process of providing specific solutions that solve security problems, thus minimizing the impact of the attacks against IS, and bearing in mind that the majority of problems take place in a similar way in different contexts, the generic solutions to these problems can be expressed as patterns [7]. These patterns will provide the methodology with great value, because they offer validated, tested and reusable solutions, whilst simultaneously compiling the knowledge of security experts [8].

Various proposals currently follow this approach, i.e., offer a systematic process for the construction of secure IS by using patterns. For example, in [11-13], the authors apply security patterns through a secure IS development method based on hierarchical architectures whose layers define the scope of each security mechanism. The main advantages of these works, which are the evolution of the same approach, are the following: in each stage, they offer the user guidelines to indicate where to apply and how to select the appropriate security pattern to satisfy the functional requirements or restrictions involved in each stage; and, they offer guidelines to identify vulnerabilities and threats in the system, along with selecting the patterns with which to mitigate them at each architectural level and at each development stage. According to these authors, one of their future works will be the implementation of this proposal in real environments. In [14], the authors put forward a method with which to integrate security patterns into a software engineering process. This proposal helps experts to close the breach between the abstract solution described in the pattern and the implementation proposed in the application. The cataloguing of different roles and the use of tools that support the systematic process make this proposal a valuable approach for real and complex organizations. However, the complexity and dynamism of these kinds of entities require an in–depth study of the detailed definition of the additional specific security tasks that are parallel to the software development in order to achieve secure IS.

After analyzing some of the proposals existing in the literature that is focused on the development of secure IS through the use of patterns, we believe that it is necessary to enrich this type of methodologies with a real and practical approach that encourages their use in a simple and systematized manner at the time of creating secure IS within real and complex organizations. This enrichment can be achieved through the detailed specification of the subjacent activities of each of the proposed stages, the elements involved and the roles taking part in each of them. It would thus be possible to provide security engineers with step by step guidelines when they confront new projects in which the existing IS within an organization need to be modified, thus mitigating the

 ISBN: 978-1-61208-158-8

errors and threats during the first stages of development of these IS and specifying the most appropriate security techniques with which to perform each of the activities and thereby maintaining the security level achieved.

We are therefore working on a methodology with which to build secure IS supported by patterns whose main objective is to offer security engineers a systematic process to be used together with the traditional software development methodologies. This will permit the construction of secure IS or maintain the security level attained in an organization's IS.

A first version of this has been published in [22], and its main characteristics are: it is based on the same stages as the classic development cycles in which we present the input and output artifacts that represent the information that is produced, modified or used for a process; the main roles taking part in each activity; and, the detailed activities of which each of the stages is composed. Another of the main contributions of this methodology is that it is focused on a central axis, which is the criticality of the assets to be protected. The use of security patterns is a fundamental contribution of our methodology since they provide structured, validated and reusable security knowledge, offering guidelines for the construction and evaluation of secure IS [9]. Finally, we would like to emphasize that this methodology is in the process of being implemented in a financial entity, and that interesting results are being obtained, which will allow us to refine, test and validate it.

In this proposal, we shall present a summary of the aforementioned methodology, whose new features consist of the in–depth study of the analysis stage in which we detail the input and output artifacts, the main roles taking part in this stage and the main activities of which it is composed. We shall support our presentation with graphical charts that represent the formalization of this methodology in SPEM (Software & Systems Process Engineering Metamodel) version 2.0 [10].

The remainder of this paper is organized as follows: Section II shows a summary of the aforementioned methodology. Section III provides a detailed description of the analysis stage. Section IV shows the current state of the application of our methodology to a real organization in the banking sector. We finish with some conclusions in Section V.

II. OVERVIEW OF THE SECURE SYSTEM DEVELOPMENT METHODOLOGY

In this section, we shall present a summary of the methodology on which we are working.

This methodology is intended to be used in parallel with and in addition to traditional software development methodologies with the purpose of strengthening the IS development process, along with being able to guarantee security against attacks and threats that place the confidentiality, availability and integrity of the assets located in these systems in jeopardy. It is also based on methodologies such as the Unified Process [15] in which a development and implementation process is carried out in an iterative and incremental manner. The advantage of this type of processes lies of the fact that we can perform successive refinements to identify risks and security critical errors during the first stages by using test mechanisms during each one of these stages to obtain a final effective and optimum solution. The structure of this systematic process follows the classic software development cycle in stages, and the main characteristics of each stage are as follows:

Analysis Stage: Set of activities centered on the achievement of security requirements according to a proposed business model. After the achievement of these requirements, a feasibility study is carried out that is focused on the assets to be protected. In this study, we analyze the risks and threats that may affect the organization's IS, and the technical possibilities with which to tackle the proposed solution in the form of security architecture.

Design Stage: Set of activities centered on the design of the final security technological solution that mitigates the risks and threats detected in the previous stage. Here, we select the structural technological elements of which the IS will be composed. We also plan the process of construction of this system in detail, and identify the tasks and personnel that will be in charge of carrying them out to obtain the proposed security architecture. All this design is focused on the criticality of the organization's assets that must be protected.

Construction Stage: The IS proposed in the previous stage will be built in this stage, and the necessary maintenance security patterns are simultaneously defined in order to guarantee the reliability and maintainability of the model built.

Test Stage: After integrating the system hardware and software components, it is necessary to guarantee their correct functioning and that they satisfy the needs indicated in the previous stages before delivering the system to the end user. The test security patterns that will show the security tests to be carried out in the future in the IS built are also defined in this stage.

Maintenance Stage: Set of specific activities that are periodically executed to guarantee that the level of security attained has not diminished over time with the appearance of new threats or security risks, or new needs not only of the end user but also of the organizational environment.

III. SECURE SYSTEM ANALYSIS

This section will show details of the analysis stage, describing its input and output artifacts together with the main roles and activities taking part in it.

The main objective of this stage is to carry out an iterative and incremental process to detect security risks that may affect the organization if the proposed business model is implemented, along with an in-depth analysis of the impact that it could have on the organization's IS.

1) Artifacts: The concept of artifact will be used as a piece of information that is produced, modified or used by a process [16]. We shall now detail the artifacts that will

 ISBN: 978-1-61208-158-8

be involved in this stage. Figure 1 shows a UML diagram containing each of these artifacts and the relations between them.

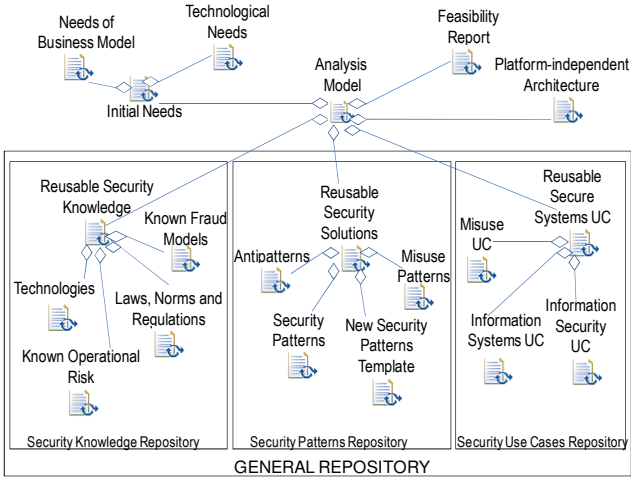


Figure 1. Analysis Stage Artifacts.

- *Initial Needs*: the input artifact for this stage, which defines the initial needs and requirements that the stakeholders need that are covered by the system. It will be composed of: the *Needs of Business Model* artifact, which specifies the business functional requirements, i.e., the initial needs required by the customer; and the *Technological Needs* artifact, which is the translation of the customer needs into technical requirements that will be used to obtain the needs at the technological infrastructure level, which must be supported by the organization's IS that cover the functional needs, the security needs and the needs of the organizational environment.
- *General Security Repository* artifact (Input/ Output): This is the most relevant artifact, both in this activity and in the whole methodology in general. It consists of an innovative element that allows this methodology to be used in real cases within complex organizations. This is owing to the fact that it is composed of several security specific repositories that collect the accumulated knowledge on this matter in different ways (real cases databases, security patterns and security use cases (UC) extracted from real IS). This artifact can also be reusable, thus minimizing the effort needed by the engineers in charge of this task to obtain validated, tested and secure IS. It will always be updated with the feedback from each of the new projects analyzed. The *General Security Repository* is formed of the following artifacts:
  - *Security Knowledge Repository*: artifact composed of different databases that compile advanced knowledge in the field of information security. This includes the *Known Fraud Models,* which is a specific repository that contains information related to known technological fraud events in the sector in which the organization operates, together with compensatory measures used to mitigate the attacks. This database will be fed by experts in the field of fraud and technological crimes based on their own experience and knowledge, and on other real data sources such as OWASP and SANS [17, 18]. The business model needs, the technical needs and the functional needs will be associated with the known fraud models to perform a study of the threats that the implementation of the model implies; *Known Operational Risk*, which is formed of a knowledge database with operational risk models and compensatory measures in relation to this kind of risk. We shall verify whether the proposed business model includes the risk of losses resulting from a lack of adaptation or from a failure in the processes, personnel or internal systems, or as a result of external events; *Laws, Norms, Regulations* collected as a repository that contains the legal restrictions that may be imposed by the country in which the organization's IS are located, the regulatory restrictions of the sector in which the organization operates and the organization's own rules in each particular project. This information will be used to carry out a study to certify that the business model does not breach any of these aspects; and finally, *Security Technologies*, which are grouped as a repository that will be verified with the business needs to define, which products/ technologies are the most appropriate to cover the needs and security risks of the proposed business model.
  - *Reusable Patterns Repository*: The artifact will include: patterns similar to *Misuse Patterns*, which relate possible attacks or misuses to the security measures that mitigate them [7]; *antipatterns* [19, 20]: *security patterns* with the structure shown in [21] that contain three levels of solutions for a specific security problem; and, *traditional security patterns* [8]. These patterns will be used to link the business model requirements, the associated security problems and the misuses that can be derived to solutions, which have already been validated and tested.
  - *Security UC Repository*: a reusable artifact that represents IS use case diagrams (UC, actors and relations), describing their behavior and capturing the requirements needed to develop a secure IS. The purpose of this artifact is to provide an IS overview through UC diagrams, capturing the main security characteristics of this kind of systems. It will be composed of other artifacts defined in the repository, which are validated and tested solutions that will assist us to improve and reduce the time and effort needed in the analysis stage. This artifact will in turn be composed of the *Reusable Secure System UC* diagrams artifact

 ISBN: 978-1-61208-158-8

that defines use case diagrams for secure IS that have been built to define common scenarios and behaviors associated with this kind of systems. This reusable artifact defines generic UC diagrams, which have been built or defined in other developments and, which are useful for this application because they contain common aspects that do not vary from one IS to another. These diagrams could also be merged with other more complex UC diagrams to represent the final IS. The *Reusable Secure System UC* artifact represents the reusable use cases, the actors and the relations between them, in order to obtain a secure IS. It is formed firstly of the *Information System UC artifact* that represents different use cases defined within an IS, which could be new use cases defined for a specific IS or reusable use cases from the repository and, which represent common functionality and technological requirements for this kind of IS. It is secondly formed of the *Information Security UC* artifact, which is similar to the previous artifact but with the difference that it captures security aspects from IS. It is in turn composed of *Security UC* and *Misuse UC,* which show security behaviors in this kind of environments, identifying possible threats and attacks against the IS itself or against the assets that must be protected, in addition to defining appropriate security requirements with which to mitigate them.

- *Analysis Model* (Output): Set of elements that are the result of the execution of the different activities in this stage. This artifact will contain the summary of the tasks developed, i.e., the initial requirements, the technological and security needs, the possible risks, threats, and legal restrictions, the security patterns identified, along with the misuse patterns, and antipatterns that are associated with the proposed business model. In addition, as output elements with own entity within this artifact, we can find the *Feasibility Report,* which is an output artifact that certifies the performance of the feasibility analysis carried out by those in charge of IS security within the organization. It presents the elements of the analysis model and its aim is to be evaluated by the relevant departments that must decide whether or not to implement the proposed business model; and finally, the *Platform- Independent Architecture* output artifact [21], which contains a high level architecture that provides a description of the security functionalities that the IS should have, independently of its technological characteristics and implementation details. More specifically, it is a conceptual description of the security mechanisms that should be incorporated into the organization's IS according to the proposed model, together with the type of relations that exist between them to guarantee the security of the organization's IS.

2) Main Roles: We shall now specify the main roles that will take part in the analysis stage, along with the functions to be developed by them (see Figure 2 in SPEM 2.0). We would like to stress that some of these roles can be executed by the same person or group of people in certain organizations.

- *Project Manager*: Role in charge of leading the project development with specific knowledge of management, and whose responsibility it is to coordinate the different security groups to obtain the performance of the Project. He will organize and supervise the Analysis Stage.
- *Security Requirements Engineer*: Role in charge of the collection of the requirements according to the proposed business model. He must be able to translate the business model needs into the technological language, extracting the main security issues that the performance of the Project implies.
- *Risk Analyst***:** In charge of leading and organizing the risk analysis related to the proposed business model. He must coordinate this task by, on the one hand studying whether the operative risk analysis will be able to detect this type of risk linked to the achievement of the business model and, on the other hand, managing the preventive analysis that will be supported by the security expert in the field of fraud and technological crimes, in addition to managing the legal analysis that will be supported by the Legal Consultant in this case.
- *Fraud Analyst*: Person in charge of leading the analysis to avoid fraud and possible technological crimes associated with the proposed business model. This stakeholder should provide current knowledge regarding fraud tendencies, new attacks, and compensatory measures to mitigate them.
- *Legal Consultant*:. Support personnel for the risk analyst who will carry out the evaluation of the risks that are inherent in the legal material with regard to existing laws, rules and regulations concerning privacy and information security.
- *Security Analyst*: Person in charge of leading and coordinating the analysis of the security requirements obtained by the *Security Requirements Engineer*. He will also be in charge of analyzing the security threats, misuses, etc., along with describing the technological security needs of the solution.
- *Security Expert*: Role specialized in determined security fields that will help the security team in very specific tasks in which the permanent members of this team do not have the necessary knowledge. He will give advice about necessary new products, proposed technologies with which to carry out the solution model, specialized tools for specific tasks, etc.
- *Security Architect*. In charge of designing the security technical architecture according to the technological and security requirements with the purpose of guaranteeing the security of the organization's IS. The

 ISBN: 978-1-61208-158-8

infrastructure designed will be implemented later, in the following stages of the methodology.

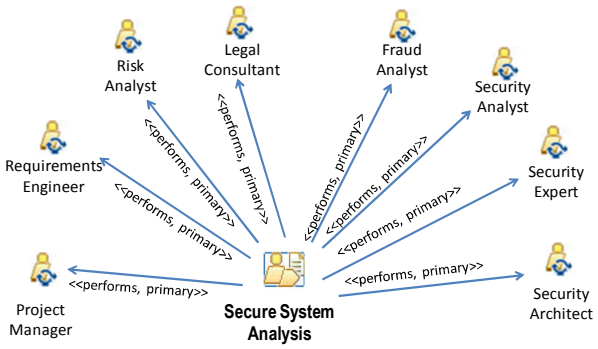


Figure 2. SPEM 2.0. view of Stakeholders

3) Analysis Stage activities: The aforementioned internal artifacts are produced in this stage. In some cases, they are the output artifacts of some activities and the input artifacts of others. The Analysis Model artifact that will serve as an input artifact for the Design stage and will certify the thorough security analysis of the proposed model will eventually be composed of all these artifacts. The main activities that will be carried out in this stage are detailed below:

- Identifying Security Systems UC: An analysis of the initial needs of the proposed business model is performed in this activity (see Figure 3). Once these needs are known, we identify the assets that must be protected and we carry out the risk analysis according to known fraud cases, operational risk associated with this business model, internal rules of the organization, regulations in the sector in which the organization operates, and laws that may affect the solution depending on where the organization's IS are located. This analysis is performed to identify the *Security UC* and *Misuse UC* that apply to the business model, and these UC can be collected in the *Security UC Repository* or can be defined by the user when a new project is analyzed. The repository will be fed back with the UC of the business model analyzed.

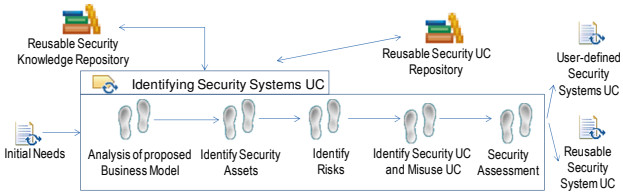


Figure 3. Activities to identify Security Systems UC in SPEM 2.0.

- Identifying Security Patterns: An association between the *Security UC* and the *security patterns* that solve the security needs specified in these use cases will be carried out. We shall also analyze the antipatterns associated with the Misuse UC to avoid security risks that may affect the solution in later stages. We shall additionally identify the technological requirements that will be mapped with the template presented in [21] to obtain the *Platform- Independent Architecture* output artifact. Finally, we shall create the *Analysis Model* that will be refined to check that there are no new security risks or technological needs, to eventually use it as an input artifact in the Design stage. Figure 4 shows this set of activities.

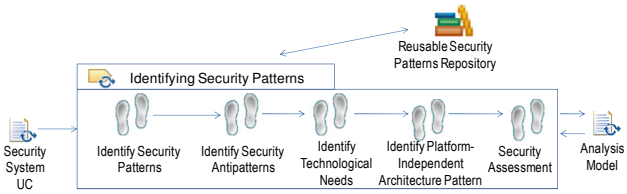


Figure 4. Activities to identify Security Patterns in SPEM 2.0

IV. DISCUSSION: CURRENT STATE OF THE METHODOLOGY

In real organizations' IS whose complexity increases daily, security aspects continue to be non-functional requirements within the software development process. This situation signifies that security aspects are, in most cases, detected in the final stages of IS construction or even when the IS is already working, thus increasing the cost and time spent on modifying the IS produced.

The experience of implementing a systematic process, which is parallel and additional to the software development process in a real organization is providing interesting results. First of all, we have observed that the different teams taking part in IS development are more and more interested in being advised by security teams in the first stages of the process, to be guided in relation to how to design the IS to avoid security threats and inherent risks. This is owing to the fact that this collaboration becomes an objective, foreseen and reliable participation, which encourages the other groups to involve the security team in all the changes that occur in the organization. Time and costs are thus saved because these changes are made during the first stages of the project, and the organization is benefited. We should also mention other benefits such as: Homogeneity between the means of working in the organization and the IS built, not only in the main headquarters but also in the acquired external entities; efficiency at the time of confronting new projects because a systematic process is available to manage each of the steps involved in building or maintaining a secure IS; cataloguing of all the provided security solutions as patterns, whose main value is to allow a fast localization and modification of IS against a threat or a suffered or foreseen risk, or its agile optimization; and, a great exportability of the means of working to any new entities acquired.

V. CONCLUSION AND FUTURE WORKS

Our research line is centered on developing a methodology for the construction of secure IS based on

 ISBN: 978-1-61208-158-8

patterns with the aim of helping security engineers in the creation of secure IS or in the maintenance of the security level attained within the IS of a real and complex organization. The systematic process, which we are working on is based on traditional development methodologies, including their key stages and dividing each of these stages into clearly defined activities that will guide engineers when adding security to an IS. In each of the stages, we show the input and output artifacts that represent the initial elements of each stage and the results, which we expect from each one of them, together with the ideal roles to develop each activity. The use of security patterns provides us with agility when solving security problems because these kinds of solutions compile the knowledge of security experts and are already validated and tested solutions that solve common security problems.

In this work, we have shown a summarized general overview of each of the stages of the methodology, and we have provided an in-depth study of the analysis stage, detailing its input and output artifacts, the roles taking part in it and the main activities of which it is composed. This presentation is supported by the formalization of the methodology in a metamodelling language (SPEM 2.0.), which has been validated and approved by the scientific community.

Finally, we should like to emphasize that the methodology proposed herein is being used in the implementation stage in a large financial entity, and this is providing us with interesting results that will help us to refine and validate it.

In future works, we shall carry out an in-depth study of the remaining stages, in addition to presenting practical examples that will certify their use in a real and complex organization. We are also working on another line consisting of building a tool to support the whole process which will serve to control each of the activities, artifacts, and people taking part in the construction of a secure IS within a real and complex organization.

ACKNOWLEDGMENT

This research has been carried out in the framework of the following projects: MODEL-CAOS (TIN2008-03582/TIN) financed by the Spanish Ministry of Education and Science, SISTEMAS (PII2I09-0150-3135) and SERENIDAD (PEII11-0327-7035) financed by the “Viceconsejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha” and the FEDER, and BUSINESS project (PET2008-0136) financed by the "Ministerio de Ciencia e Innovación”, Spain.

REFERENCES

[1] S. T. Halkidis, N. Tsantalis, A. Chatzigeorgiou, and G. Stephanides: "Architectural Risk Analysis of Software Systems Based on Security Patterns," IEEE Transactions on Dependable and Secure Computing 5, pp. 129-142, 2008.

[2] R. E. Andrew, A. P. Moore, L. Bass, M. Klein, and F. Bachmann: "Security and Survivability Reasoning Frameworks and Architectural Design Tactics," SEI, 2004.

[3] J. Jürjens: "Secure Systems Development with UML," Springer-Verlag, 2004.

[4] H. Mouratidis and P. Giorgini: "Integrating Security and Software Engineering: Advances and Future Vision," IGI Global, 2006.

[5] R. Pressman: "Software Engineering: A Practitioner's Approach," McGraw-Hill Science/Engineering/Math, 2004.

[6] T. Roberts: "Why can't we implement this SDM?," IEEE Software 16, pp. 70 - 71, 1999.

[7] E. B. Fernandez, N. Yoshioka, and H. Washizaki: "Modeling Misuse Patterns," International Conference on Availability, Reliability and Security (ARES '09), pp. 566-571, 2009.

[8] M. Schumacher, E. B. Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad: "Security Patterns: Integrating Security and Systems Engineering," 2006.

[9] R. Ortiz et al.: "Applicability of Security Patterns," The 5th International Symposium on Information Security (IS'10 - OTM'10), Crete, Greece, 2010.

[10] OMG: "Software & Systems Process Engineering Meta-Model Specification (SPEM) 2.0," 2008.

[11] E. B. Fernandez, M. M. Larrondo-Petrie, T. Sorgente, and M. VanHilst: "Chapter 5, A methodology to develop secure systems using patterns," Integrating security and software engineering: Advances and future vision, IDEA Press, pp. 107-126, 2006.

[12] E. B. Fernandez: "Security Patterns and A Methodology to Apply them," Security and Dependability for Ambient Intelligence, pp. 37-46, 2009.

[13] E. B. Fernandez et al.: "Using security patterns to develop secure systems," H. Mouratidis (ed.): Software Engineering for Secure Systems: Industrial and Research Perspectives, pp. 16-31, 2009.

[14] F. Sanchez-Cid and A. Maña: "SERENITY Pattern-Based Software Development Life-Cycle," 19th International Workshop on Database and Expert Systems Application (DEXA '08), pp. 305-309, Turin, 2008.

[15] P. Kruchten: "The Rational Unified Process: An Introduction," Addison-Wesley, Boston, 2000.

[16] D. G. Rosado, E. Fernández-Medina, J. López, and M. Piattini: "Analysis of Secure Mobile Grid Systems: A systematic approach," Information and Software Technology, 2010.

[17] The Open Web Application Security Project (OWASP) <http://www.owasp.org> 01.04.2011

[18] SANS - Computer Security Training, Network Research & Resources <http://www.sans.org> 05.04.2011

[19] M. Kis: "Information Security Antipatterns in Software Requirements Engineering," 9th Conference of Pattern Languages of Programs, 2002.

[20] J. Král and M. Zemlicka: "Popular SOA Antipatterns," Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, pp.271-276, Athens, Greece, 2009.

[21] S. Moral-García et al.: "A New Pattern Template to Support the Design of Security Architectures," The Second International Conferences of Pervasive Patterns and Applications, Lisbon, Portugal, 2010.

[22] R. Ortiz, S. Moral-Rubio, J. Garzás, and E. Fernández-Medina: "Towards a Pattern-Based Security Methodology to Build Secure Information Systems," 8th International Workshop on Security in Information Systems, Beijing, China, 2011.

 ISBN: 978-1-61208-158-8