SERVICE COMPUTATION 2017

The Ninth International Conferences on Advanced Service Computing

ISBN: 978-1-61208-528-9

February 19 - 23, 2017

Athens, Greece

SERVICE COMPUTATION 2017 Editors

Marcelo De Barros, Microsoft Corporation - Seattle, USA

Janusz Klink, Wroclaw University of Science and Technology, Wroclaw, Poland

Tadeus Uhl, Flensburg University of Applied Sciences, Germany | Maritime University of Szczecin, Poland

Thomas Prinz, University of Jena, Germany

SERVICE COMPUTATION 2017

Forward

The Ninth International Conferences on Advanced Service Computing (SERVICE COMPUTATION 2017), held between February 19-23, 2017 in Athens, Greece, continued a series of events targeting service computation on different facets. It considered their ubiquity and pervasiveness, WEB services, and particular categories of day-to-day services, such as public, utility, entertainment and business.

The ubiquity and pervasiveness of services, as well as their capability to be context-aware with (self-) adaptive capacities create challenging tasks for services orchestration, integration, and integration. Some services might require energy optimization, some might requires special QoS guarantee in a Web-environment, while other a certain level of trust. The advent of Web Services raised the issues of self-announcement, dynamic service composition, and third party recommenders. Society and business services rely more and more on a combination of ubiquitous and pervasive services under certain constraints and with particular environmental limitations that require dynamic computation of feasibility, deployment and exploitation.

The conference had the following tracks:

- Service measurement and evaluation
- Service quality
- Advanced Analysis of Service Compositions
- Challenges

We take here the opportunity to warmly thank all the members of the SERVICE COMPUTATION 2017 technical program committee, as well as all of the reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors that dedicated much of their time and effort to contribute to SERVICE COMPUTATION 2017. 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 also gratefully thank the members of the SERVICE COMPUTATION 2017 organizing committee for their help in handling the logistics and for their work that made this professional meeting a success.

We hope that SERVICE COMPUTATION 2017 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in the field of advanced service computing. We also hope that Athens, Greece provided a pleasant environment during the conference and everyone saved some time to enjoy the charm of the city.

SERVICE COMPUTATION 2017 Committee

SERVICE COMPUTATION 2017 Steering Committee

Mihhail Matskin, KTH, Sweden
Bernhard Hollunder, Hochschule Furtwangen University – Furtwangen, Germany
Paul Humphreys, Ulster Business School/University of Ulster, UK
Arne Koschel, Hochschule Hannover, Germany
Michele Ruta, Technical University of Bari, Italy
Alfred Zimmermann, Reutlingen University, Germany
Aida Omerovic, SINTEF, Norway
Annett Laube, Bern University of Applied Sciences (BUAS), Switzerland
Claus Pahl, Dublin City University, Ireland

SERVICE COMPUTATION 2017 Industry/Research Advisory Committee

Marcelo De Barros, Microsoft, USA
Steffen Fries, Siemens Corporate Technology - Munich, Germany
Matthias Olzmann, noventum consulting GmbH - Münster, Germany
Rong N. Chang, IBM T.J. Watson Research Center, USA
Jan Porekar, SETCCE, Slovenia

SERVICE COMPUTATION 2017

Committee

SERVICE COMPUTATION Steering Committee

Mihhail Matskin, KTH, Sweden
Bernhard Hollunder, Hochschule Furtwangen University – Furtwangen, Germany
Paul Humphreys, Ulster Business School/University of Ulster, UK
Arne Koschel, Hochschule Hannover, Germany
Michele Ruta, Technical University of Bari, Italy
Alfred Zimmermann, Reutlingen University, Germany
Aida Omerovic, SINTEF, Norway
Annett Laube, Bern University of Applied Sciences (BUAS), Switzerland
Claus Pahl, Dublin City University, Ireland

SERVICE COMPUTATION 2017 Industry/Research Advisory Committee

Marcelo De Barros, Microsoft, USA
Steffen Fries, Siemens Corporate Technology - Munich, Germany
Matthias Olzmann, noventum consulting GmbH - Münster, Germany
Rong N. Chang, IBM T.J. Watson Research Center, USA
Jan Porekar, SETCCE, Slovenia

SERVICE COMPUTATION 2017 Technical Program Committee

Antonia Albani, Institute of Information Management | University of St. Gallen, Switzerland
Irina Astrova, Tallinn University of Technology, Estonia
Jocelyn Aubert, Luxembourg Institute of Science and Technology (LIST), Luxembourg
Souvik Barat, Tata Research Development and Design Center, India
Carlos Becker Westphall, Federal University of Santa Catarina, Brazil
Ali Beklen, HotelRunner, Turkey
Kadda Beghdad Bey, EMP, Algiers, Algeria
Juan Boubeta-Puig, University of Cádiz, Spain
Antonio Brogi, University of Pisa, Italy
Massimo Cafaro, University of Salento, Italy
Oscar Mauricio Caicedo Rendon, University of Cauca, Brazil
Isaac Caicedo-Castro, University of Córdoba, Colombia
Juan-Carlos Cano, Universidad Politecnica de Valencia, Spain
Wojciech Cellary, Poznan University of Economics, Poland
Chin-Chen Chang, Feng Chia University, Taiwan
Rong N. Chang, IBM T.J. Watson Research Center, USA
Dickson Chiu, The University of Hong Kong, Hong Kong
Marcelo De Barros, Microsoft, USA

Table of Contents

A Study of the InnoDB Storage Engine in the MySQL 5.6

Hong-Jin, Park
School of Computer, Information and Communications
Sangji University
83 Sangjidae-gil, Wonju, Gangwondo, Korea
hjpark1@sangji.ac.kr

***Abstract*—MySQL is the most widely used open source relational database management system. The logical structure of MySQL is largely divided into three layers and the storage engine belongs to the last layer. The purpose of this paper is to evaluate the performance of InnoDB engine, the engine that supports transaction of several storage engines used in MySQL 5.6. TPC-C (Transaction Processing performance Council – C), an international standard model of benchmarking, is used for performance evaluation and performance evaluation was carried out by installing HammerDB for this purpose. It was found that, if the number of user requests increases and the number of transaction processing requests increases, then the number of transactions processed in InnoDB also increases.**

Keywords- InnoDB, Storage Engine, TPC-C, MySQL 5.6

I. INTRODUCTION

A database management system (DBMS) is used when many users deal with a large amount of data in large computer systems. In particular, the relational database management system is a system for managing data to be stored in a two-dimensional data format composed of rows and columns. Generally, data on the server can be manipulated and modified according to commands written in SQL. Even now, several databases such as MySQL, PostgreSQL are used because the reliability is high and data classification, sorting and navigation speed are fast. Many relational database systems showing high evaluative performance are used and MySQL is the database system the most widely used universally as an open source among them.

The logical structure of MySQL database system consists of three layers. From these, the storage layer, which is the last layer, plays a role in storing and retrieving data into MySQL. MySQL 5.6 has 9 storage engines and the InnoDB engine is the one which is the most suitable for transaction processing among them. Compared to other engines, InnoDB engine provides a variety of functions such as excellent performance, automatic failure recovery function, concurrency support, various optimization supports as well as transaction processing. Currently, the InnoDB engine performs as MySQL's default engine.

This paper evaluates the performance of the InnoDB engine, which shows the best performance when processing transaction from several storage engines in MySQL 5.6. For performance analysis, we used TPC-C, one of benchmark models announced by Transaction Processing Performance Council (TPC). TPC-C is one of international standard models used as a performance benchmark for databases through Online Transaction Processing (OLTP) work. Ubuntu 14.04 LTS (Long Term Support) was used as an operating system and HammerDB was installed to evaluate performance with TPC-C.

II. MYSQL LOGICAL STRUCTURE : INNODB

MySQL is an open source relational database management system currently distributed by Oracle. The logical structure of MySQL is largely divided into three layers. There is a Handing layer in the highest layer and processes DB connection, authentication and security. Parsing of query, analysis, review, optimization and caching are performed in the middle layer of the logical structure in MySQL. There is a storage engine in the last layer and play a role of searching queried data in MySQL.

A. InnoDB Storage Engine

MySQL 5.6 provides a total of nine storage engines such as CSV, MRG_MYISAM, MyISAM, MEMORY, InnoDB etc. The available storage engines in MySQL can be checked with MySQL command SHOW ENGINES\G. MyISAM and InnoDB are the most widely used storage engines among them. The functions of MyISAM are simple and the speed for data search is faster than InnoDB. However, since the transaction to ensure safety for a series of operations performed at a time in the database is not supported, InnoDB engine is used if the data integrity should be ensured. InnoDB engine is adopted as a default engine from MySQL 5.5 and can process transactions. The InnoDB engine is also widely used in situations when the transaction does not necessarily need the best performance or an automatic failure recovery function among storage engines supporting transactions. InnoDB engine uses MVCC (MultiVersion Concurrency Control) to increase concurrency and performs a variety of optimization internally. Adaptive hash index is used to make the inquiry faster and functions to bring data predicted to be needed on the disk in advance; the Insert buffer is used to make the insertion faster.

III. PERFORMANCE EVALUATION

In this paper, after installing MySQL 5.6 on Ubuntu, we selected InnoDB as a storage engine for evaluation. In order to evaluate the InnoDB performance, HammerDB 2.19 was installed and the evaluation was carried out by using TPC-C. TPC defined a benchmark for database and transaction processing as benchmark test models. These models were announced by the transaction processing performance council and these are used to measure the performance of systems such as Disk I/O, etc. The TPC benchmark is meant to comprehensively evaluate the hardware performance including network and software performance, as well as operating system (OS) performance.

In this paper, we used HammerDB as the tool to evaluate the performance with TPC-C and the evaluation settings we used are as follows. After installing MySQL 5.6 on Ubuntu, we measured TPM (transactions per minute) values while gradually increasing virtual users from 2 people to 10 people and performing TPC-C. The performance evaluation is shown in Fig. 1 and Fig. 2.

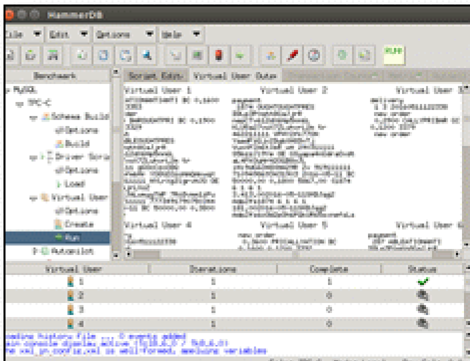

Figure 1. The View of Performance Evaluation – 1

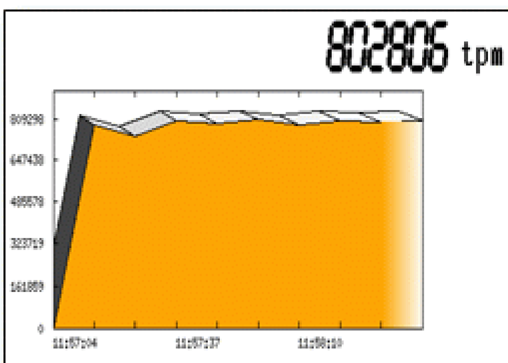

Figure 2. The View of Performance Evaluation – 2

Fig. 3 shows MySQL InnoDB Engine Performance evaluation. TPM is shown while gradually increasing the number of users from 2 people to 10 people. First, it can be seen that, if there are two users, 668,112 transactions are processed per minute and if users are 10 people, 802,806 transactions are processed per minute. If based on 2 users, transactions per minute were processed more quickly by 9% in the case of 4 people than in the case of 2 people and the processing speed was faster by approximately 12%, 18%, 20% in the case of 6 people, 8 people, 10 people, respectively. That is, the processing speed was faster by 20% when each of 10 users requests 10,000 transactions, than when each of 2 users requests 10,000 transactions and this shows that transaction processing is a lot within the same time.

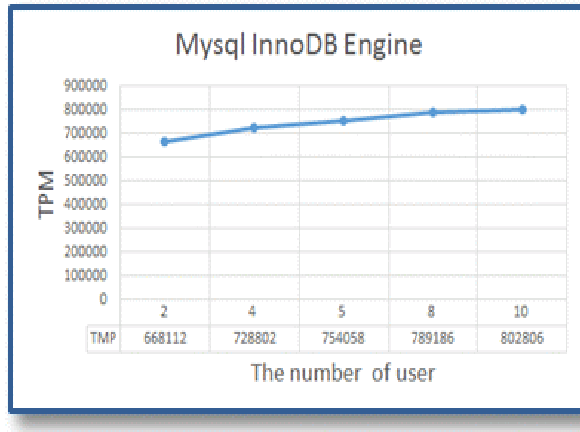

Figure 3. MySQL InnoDB Engine Performance

IV. CONCLUSION

In this paper, after installing MySQL 5.6 on Ubuntu, we selected InnoDB as a storage engine for evaluation. In order to evaluate the InnoDB performance, HammerDB 2.19 was installed and the evaluation was carried out by using TPC-C. It was found that, if the number of user requests increases and the number of transaction processing requests increases, then the number of transactions processed in InnoDB also increases.

REFERENCES

[1] Cornelia Győrödi, Robert Győrödi , George Pecherle, Andrada Olah, “A comparative study: MongoDB vs. MySQL”, 13th International Conference on Engineering of Modern Electric Systems (EMES), 2015
[2] Letkowski, Jerzy, “Doing database design with MySQL”, Journal of Technology Research, Jan. 2015
[3] Rafael Almeida, Pedro Furtado, Jorge Bernardino, “Performance Evaluation MySQL InnoDB and Microsoft SQL Server 2012 for Decision Support Environments”, Journal of the Web Technologies and Applications, Nov. 2015
[4] Yong Chen, Xuanzhong Xie, Jiayin Wu, “A Benchmark Evaluation of Enterprise Cloud Infrastructure”, Proceedings of the Eighth International Conference on Computer Science & Software Engineering, 2015
[5] Jae-Hwan Lee, Young-Hun Lee, Jong-an park, “Performance Analysis Measurement of Storage according to INNODB Parameter Change”, International Journal of Software Engineering and Its Applications, 2013
[6] http://dev.mysql.com/doc/refman/5.6/en/ [accessed January 2017]

Structuring Software Fault Injection Tools for Programmatic Evaluation

Lukas Pirl, Lena Feinbube, and Andreas Polze
Operating Systems and Middleware Group
Hasso Plattner Institute at University of Potsdam
Email: lukas.pirl@student.hpi.de, {lena.feinbube, andreas.polze}@hpi.de

***Abstract*—The increasing complexity of software systems challenges the assurance of the likewise increasing dependability demands. Software fault injection is a widely accepted means of assessing dependability, but far less accessible and integrated into engineering practices than unit or integration testing. To address this issue, we present a dataset of existing fault injection tools in a programmatically evaluable model. Our Fault Injection ADvisor (FIAD) suggests applicable fault injectors mainly by analyzing definitions from Infrastructure as Code (IaC) solutions. Perspectively, FIAD can yield findings on how to classify fault injectors and is extensible in a way that it can additionally suggest workloads or run fault injectors.**

***Keywords*–fault injection; infrastructure as code; testing; service-oriented systems; distributed systems.**

I. Introduction

Facilitated by the advent of cloud computing, our society – politically, commercially and individually – increasingly relies on utility and service computing. The need for growing quality and quantity of such ubiquitous distributed software systems boosts their complexity. There is thus a dire need for dependability assessment of such systems but they are notably hard to test: Complex software systems fail in complicated ways [1] even when fault tolerance is implemented [2]. It is widely recognized that *Software Fault Injection (SFI)* should be part of the development process [3]. Yet, SFI lacks the high grade of automation and adaption of testing. Many fault injectors remain research prototypes, others are used only in high criticality systems.

We assume and address a practical issue with SFI tools (*injectors*): Knowledge about available injectors is not user-friendly accessible and it is thus hard to find applicable ones for a given infrastructure. We elaborate on a dataset consolidating the aforementioned knowledge, its programmatically evaluable model, and the automatic acquisition of information about infrastructures. Ultimately, this paper presents a tool which automatically determines applicable injectors by analyzing systems' IaC descriptions.

After core concepts have been introduced in the remainder of this section, Section II outlines related work on the topic. While Section III defines the research problem and our approach to address it on a theoretical level, the subsequent section describes our concrete realization. Section V demonstrates an example use case including characteristic code excerpts. Finally, Section VI draws conclusions from the findings of this work and proposes areas of possible further investigation.

A. Software Fault Injection

Fault injection testing is a well-established approach to test the fault tolerance of computer systems. We use the term SFI to denote the software-implemented injection of *software faults*. This is not to be confused with Software-Implemented Fault Injection (SWIFI), that mainly refers to the injection of *hardware faults*, implemented in software. In the broad field of SFI, many tools exist that differ in target applications, usability and their implementation strategy. For practitioners wanting to embed SFI in a software development process, finding suitable injectors is tedious and likely results in testing various prototypes of varying fitness and quality.

B. Infrastructure as Code

The increasing complexity of distributed systems has amplified the interest in the IaC paradigm [4]. This paradigm describes the definition of infrastructures in a machine- *and* human-readable format. Ideally, IaC enables fully automated provisioning, as well as deployment. Additionally, it has the advantageous side effect that other tools can reuse the infrastructure definitions.

Ansible [5] is an open source IT configuration management, deployment, and orchestration tool of increasing significance [6]. The desired state of the targeted infrastructure is defined declaratively in the YAML [7] file format. Using the same format, so-called *Playbooks* consolidate the desired state regarding an infrastructure at the topmost level. Enterprises, such as Apple and Juniper [8], as well as large open source projects, such as OpenStack [9], use Ansible actively.

II. Related Work

The concept of dependability [10] is an area of interest for computer scientists since decades. Recently, the awareness that the dependability of distributed systems needs in-depth assessment has pervaded major Web enterprises. Netflix injects faults into its production systems using ChaosMonkey [11], similarly to Amazon [12] and Etsy [13]. While earliest fault injectors focused on hardware, SFI became popular in the 1990s (e.g., FIAT [14]). It has been acknowledged that the dependability bottleneck is not within the hardware but within the software layers [15], [16]. Natella et al. present a comprehensive up-to-date survey of SFI approaches from research [17]. The research areas of SFI and "classical" software testing overlap, and the distinction between "workload" and "faultload" blurs. [18] presents a fault model oriented view on software testing. The approach of fuzz testing (e.g., CRASHME [19]) also exemplifies the overlap of fault injection and testing: Here, the "fault model" consists of special inputs to the same API accessed during unit testing. When cataloging injectors, it is hence imperative to consider a broad spectrum of tools from both domains. Our dataset includes popular fuzzers, such as Trinity [20], which are excluded from traditional surveys. Software testing strategies are further discussed theoretically in [18] and surveyed in [21].

Despite decades of research, there is no “cookbook for SFI testing”. To the best of our knowledge, no programmatically evaluable catalog of injectors exists.

III. Problem Statement and Approach

Distributed software systems grow in complexity, also due to the rapid rise of cloud computing. Their dependability demands grow along with their importance for enterprises and customers. Distributed software systems are notoriously difficult to test and fail even when adhering to well-established engineering practices (see Section I). SFI is a scalable and versatile approach to assess their dependability experimentally.

Unit and integration testing is widely incorporated into software development processes and allows for a decent level of automation. In contrast, SFI is yet less established and remains a research rather than an engineering topic. One reason may be that the application of SFI to an existing product currently requires significant knowledge and manual effort. It is laborious to get an overview of available injectors and to check their applicability regarding an infrastructure.

To overcome this burden, we leverage the unprecedented availability of information provided by IaC solutions. In particular, our implementation will initially be based on Ansible because of its growing popularity. Our tool – FIAD – hence analyzes Playbooks for the detection of characteristics required by injectors (hence *targets*). Further, we created a dataset of injectors from research, industry and open source projects. Combining these sources of information, FIAD programmatically identifies applicable injectors. Due to the specificity of dependability requirements, users are obliged to judge the fitness of the injectors’ fault models. Possible users include dependability researchers and actors in the software development process, such as testers, developers and engineers.

Given that an infrastructure is represented using Playbooks, this approach introduces no extra effort. Also, there are no minimal requirements regarding the information available via Playbooks but self-evidently, the more complete the Playbooks, the more accurate and complete the results. Since no access to the deployed infrastructure is required, FIAD can as well be used on infrastructures outside the own administrative domain.

Test environments and workloads often differ from the production system, which can lead to uncertainty regarding the fault tolerance of the latter [13]. Since FIAD operates on the basis of infrastructure descriptions, it is workload-independent. Hence, it can analyze the actual production system and does not suffer from the aforementioned problem.

IV. Concepts and Realization

At its core, the realized model is a two-sided object oriented class hierarchy: one for injectors and one for targets. Accompanied by the simplified illustration in Fig. 1, the following Subsections elaborate further on its details. We aimed at a programmer oriented model of injectors and targets which satisfies the requirements listed below:

1) It should allow to **flexibly classify injectors by multiple characteristics** and thereby accumulate knowledge from research and documentation of tools.
2) It should be **explicit and not overly abstract**, allowing users to easily understand, customize and extend it.
3) It should **allow for programmatic evaluation**.

FIAD is a command line interface application that reads Playbooks and outputs a list of applicable injectors. The output enables users to get an initial overview of applicable injectors or to expand their existing SFI setup. FIAD is capable of connecting to hosts referenced in Playbooks for the collection of *Facts* [22]. Facts are pieces of information about the target hosts, such as virtualization, hardware and networking details.

The internal procedure of an execution of FIAD is visualized in Fig. 2. Initially, all models (i.e., injectors and targets) register themselves at a registry and are set up in the second step. If required, FIAD detects Playbooks recursively in a specified directory. Thirdly, Playbooks are loaded from files using Ansible’s sophisticated internal mechanisms supporting Playbook dependencies, file inclusions etc. Upon explicit request of the user, FIAD gathers Facts from all hosts referenced in Playbooks. Fourthly, all loaded Playbooks are handed to all registered targets to enable the latter to detect themselves within the former. Detected targets then imply further targets accordingly. Finally, detected and implied targets are matched against injectors and the results are displayed.

FIAD is implemented in a modular structure where the topmost coordination happens within the module *cli*. Alongside with some auxiliary modules (e.g., to load and aid in accessing Playbooks and Facts), the implementations of injectors and targets are kept in the module *models*. This module encapsulates all the domain-specific knowledge and is the main extension point of FIAD’s dataset and capabilities. When adding an injector or a target, the use of a registry eliminates the need to alter code outside the corresponding module.

A. Injectors

A central contribution is a programmatically evaluable dataset of existing SFI tools (*injectors*). Previous surveys [17], [23] mainly took research in the form of published conference or journal articles into consideration. Yet, for users searching for pragmatic answers to the question of which injector to use, besides the vast body of research, practical experience of industrial SFI is relevant. Hence, our dataset of injectors consolidates knowledge from research, commercial and open source origins. Developers can additionally include new injectors easily (e.g., a in-house injector of an enterprise).

Due to the wide variety of existing injectors, only a semi-structured procedure to assemble the dataset could be followed: Initially, we collected a list of injectors by surveying research publications and by examining open source, as well as commercial products. To cover a broad range of targets, no tools were actively excluded because of their maturity or origin. For every injector found, we identified characterizing properties (e.g., “injects based on a rate”). Subsequently, we deduced more abstract characteristics (e.g., “injection trigger”). Where applicable and possible, we finally determined yet unspecified properties for all combinations of tools and characteristics.

B. Targets

Injectors typically aim at a group of applications or infrastructures sharing certain characteristics, such as “instances on Amazon EC2”. We refer to a characteristic required by an injector as *target*. Targets can be at different layers of abstraction (e.g., an operating system versus a Web API).

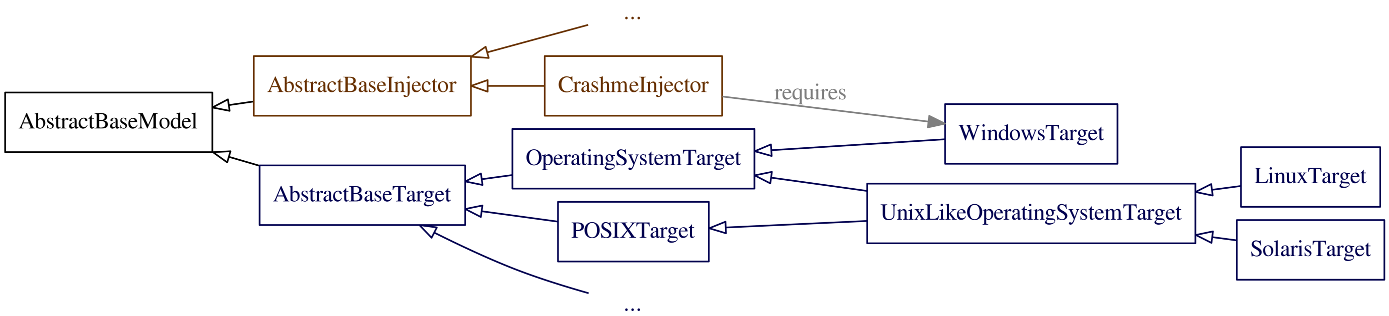

Figure 1. The class hierarchies for injectors (brown) and targets (blue). They interweave through the requirements of injectors on targets (gray).

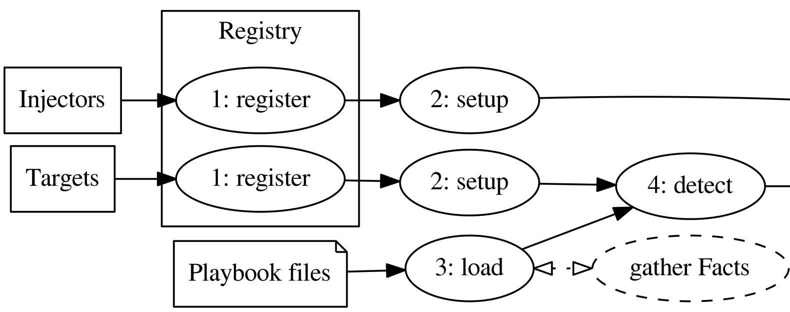

Figure 2. FIAD's execution procedure (excerpt).

Targets can be general, such as "requires a POSIX API", or more specific, such as "requires a Linux operating system". In this example, the latter target implies the former, since Linux implements the POSIX API. We use object oriented inheritance to represent such implications. Consequently, our dataset contains a hierarchy of targets as depicted in Fig. 1.

We experienced that most targets are able to detect themselves through indicating commands (C_i), Ansible modules (M_i), packages to install (P_i) or Facts gathered from hosts (F_i). As illustrated in equation (1), the default implementation considers a target detected if any of the indicators found (C_f, M_f, P_f, F_f) is present. This enables defining most targets with little effort. If a more advanced detection is required, the default can be overridden and optionally be reused selectively.

$$detected \quad \Leftrightarrow \quad \bigvee_{X \in \{C,M,P,F\}} (X_i \cap X_f \neq \emptyset) \tag{1}$$

Every detected target determines implied targets by traversing up the class hierarchy. The support for recursive traversal, taking multiple inheritance into account, allows for an even more efficient and flexible notation of targets.

C. Matching Targets with Injectors

Once all detected and recursively implied targets are known, FIAD is able to determine which injector is applicable to which infrastructure (hence a *match*). In analogy to the detection of targets, we observed that the majority of injectors share logic concerning their required targets. Again, a default behavior to determine matches is provided. It enables the definition of injectors with little effort and can be overridden and reused selectively. To use the default implementation, an injector must define a set T_{all} of targets that are *all* required to be detected, or a set T_{any} of targets of which *any* is required to be detected, or both. If both sets are provided, requirements regarding T_{any} *and* T_{all} must be fulfilled to yield a match. Equation (2) summarizes this default implementation.

$$\begin{aligned} match \quad \Leftrightarrow \quad & (\ T_{detected} \supseteq T_{all}\ ) \quad \wedge \\ & (\ (T_{any} = \emptyset) \ \vee \ (T_{detected} \cap T_{any} \neq \emptyset)\ ) \end{aligned} \tag{2}$$

V. EXAMPLE

The two following listings exemplify a fraction of the dataset of FIAD and also demonstrate the characteristically concise notation of injectors and targets. Listing 1 shows FIAD's model representing a Java Runtime Environment (JRE), Listing 2 the model representing the FATE injector [14]. Since a JRE is present when it is installed, the *JavaTarget* defines *INDICATING_PACKAGES_ANY* accordingly. In analogy, since FATE targets distributed systems written in Java, the *FATEInjector* defines *REQUIRED_TARGETS_ALL* and *REQUIRED_TARGETS_ANY* accordingly.

```
class JavaTarget(LanguageTarget):
    """
    Represents a Java Runtime Environment.
    """
    INDICATING_PACKAGES_ANY = {
        re_compile_ci(r'^jdk($|[-_][^\s]*$)'),
        re_compile_ci(r'^java($|[-_][^\s]*$)'),
    }
    # ...
```

Listing 1. The model *JavaTarget* defines a list of (regular expressions to detect) packages that indicate a JRE's presence.

```
class FATEInjector(AbstractBaseInjector):
  """
  FATE aims at a high coverage of failure scenarios
  in cloud systems, including multi-fault ones. ...
  """
  FRIENDLY_NAME = "FATE"
  PUBLICATION_BIBTEX = # ...
  WEBSITE_URL = # ...
  REQUIRED_TARGETS_ALL = {JavaTarget}
  REQUIRED_TARGETS_ANY = {EC2VMTarget,
                          AzureVMTarget, ...}
  # ...
```

Listing 2. The model of the FATE fault injector. FATE is potentially applicable if for example virtual machines in cloud environments and a JRE are detected.

We further assume actors of a software project wanting to assess the dependability of their product. To find applicable injectors effortlessly, they run FIAD on the Playbooks that exist to deploy the product. FIAD would then, for instance, find an Ansible task in which a JRE is installed as depicted in Listing 3. In accordance to the algorithm described in Section IV, the *JavaTarget* is detected and assuming the *EC2VMTarget* is detected as well, FATE emerges as a potentially promising injector. Besides suggesting FATE, FIAD can provide additional information, such as the URL of the injector's Web site or the scientific publication which described it initially.

```
- name: Install Java 1.7
  yum: name=java-1.7.0-openjdk state=present
```

Listing 3. An Ansible task which installs a JRE.

VI. Conclusion and Outlook

The presented tool FIAD provides a knowledge base to explore applicable injectors. By requiring no pre-existing experiences and by being automated, it can be integrated in software development processes and thereby raise the awareness of SFI.

The underlying model for cataloging injectors including their requirements, serves as an extensible framework and allowed us to efficiently define them as a uniform dataset. We are currently completing the dataset and hope that maintaining and updating it becomes an open source community effort. Research-wise, the dataset and its model may yield findings on how to formally classify injectors. Reusing information provided by IaC solutions for the determination of applicable injectors proved to be a powerful approach. However, further research is necessary to fully understand its versatility and possible limitations. For this purpose, an in-depth case study on OpenStack [24] is planned. By adding a new layer of abstraction, other IaC solutions could be additionally integrated, what would allow for recommendations of increased quality.

The inclusion of fault models and workloads is another compelling future extension. Filtering by injectors' fault models would help users to focus on assessing specific aspects of infrastructures' dependability. Through FIAD's capability of interacting with hosts, it could automatically apply injection campaigns and possibly even analyze their effects.

Acknowledgement & Disclaimer

This paper has received funding from the European Union's Horizon 2020 research and innovation programme 2014-2018 under grant agreement No. 644866. This paper reflects only the authors' views. The European Commission is not responsible for any use that may be made of the information it contains.

References

[1] N. R. Council, D. Jackson, and M. Thomas, Software for Dependable Systems: Sufficient Evidence? Washington, DC, USA: National Academy Press, 2007.

[2] H. S. Gunawi, T. Do, J. M. Hellerstein, I. Stoica, D. Borthakur, and J. Robbins, "Failure as a service (faas): A cloud service for large-scale, online failure drills," University of California, Berkeley, Berkeley, vol. 3, 2011.

[3] H. Madeira and P. Koopman, "Dependability benchmarking: making choices in an n-dimensional problem space," in First Workshop on Evaluating and Architecting System Dependability (EASY), 2001.

[4] M. Hüttermann, Infrastructure as Code. Berkeley, CA: Apress, 2012, pp. 135–156.

[5] Red Hat Inc., "Ansible is Simple IT Automation," visited on 2017-02-12. [Online]. Available: https://www.ansible.com

[6] RightScale, Inc., "2016 state of the cloud report," RightScale, Inc., Tech. Rep., 2016.

[7] C. C. Evans, "The Official YAML Web Site," visited on 2017-02-12. [Online]. Available: http://yaml.org/

[8] S. Ziouani, "Enterprise companies welcoming Ansible IT automation," https://www.ansible.com/blog/enterprise-ansible, Oct. 2014, visited on 2017-01-04.

[9] OpenStack, "OpenStackAnsible," visited on 2017-02-12. [Online]. Available: https://wiki.openstack.org/wiki/OpenStackAnsible

[10] J.-C. Laprie, "Dependable computing and fault-tolerance," Digest of Papers FTCS-15, 1985, pp. 2–11.

[11] A. Tseitlin, "The antifragile organization," Commun. ACM, vol. 56, no. 8, 2013, pp. 40–44.

[12] T. Limoncelli, J. Robbins, K. Krishnan, and J. Allspaw, "Resilience engineering: learning to embrace failure," Commun. ACM, vol. 55, no. 11, 2012, pp. 40–47.

[13] J. Allspaw, "Fault injection in production," Commun. ACM, vol. 55, no. 10, 2012, pp. 48–52.

[14] J. H. Barton, E. W. Czeck, Z. Z. Segall, and D. P. Siewiorek, "Fault injection experiments using FIAT," Computers, IEEE Transactions on, vol. 39, no. 4, 1990, pp. 575–582.

[15] J. Gray, "Why do computers stop and what can be done about it?" in Symposium on reliability in distributed software and database systems. Los Angeles, CA, USA, 1986, pp. 3–12.

[16] D. Oppenheimer, A. Ganapathi, and D. A. Patterson, "Why do internet services fail, and what can be done about it?" in USENIX symposium on internet technologies and systems, vol. 67. Seattle, WA, 2003.

[17] R. Natella, D. Cotroneo, and H. S. Madeira, "Assessing dependability with software fault injection: A survey," ACM Computing Surveys (CSUR), vol. 48, no. 3, 2016, p. 44.

[18] A. Pretschner, D. Holling, R. Eschbach, and M. Gemmar, "A generic fault model for quality assurance," in Model-Driven Engineering Languages and Systems. Springer, 2013, pp. 87–103.

[19] G. J. Carrette, "CRASHME: Random input testing," http://people.delphiforums.com/gjc/crashme.html, 1996, visited on 2017-01-04.

[20] D. Jones, "Trinity : A Linux system call fuzzer." visited on 2017-02-12. [Online]. Available: http://codemonkey.org.uk/projects/trinity/

[21] A. Orso and G. Rothermel, "Software testing: A research travelogue (2000–2014)," in Proceedings of the on Future of Software Engineering, ser. FOSE 2014. New York, NY, USA: ACM, 2014, pp. 117–132.

[22] Red Hat Inc., "Variables – Ansible Documentation," visited on 2017-02-12. [Online]. Available: https://docs.ansible.com/ansible/playbooks_variables.html#information-discovered-from-systems-facts

[23] H. Ziade, R. A. Ayoubi, R. Velazco et al., "A survey on fault injection techniques," Int. Arab J. Inf. Technol., vol. 1, no. 2, 2004, pp. 171–186.

[24] OpenStack, "Open source software for creating private and public clouds." visited on 2017-02-12. [Online]. Available: https://www.openstack.org/

Certification Matters for Service Markets

Marie-Christine Jakobs, Julia Krämer, Dirk van Straaten and Theodor Lettmann

Paderborn University
Paderborn, Germany
Email: {marie.christine.jakobs,juliadk,dirk.van.straaten,lettmann}@upb.de

***Abstract*—Whenever customers have to decide between different instances of the same product, they are interested in buying the best product. In contrast, companies are interested in reducing the construction effort (and usually as a consequence thereof, the quality) to gain profit. The described setting is widely known as opposed preferences in quality of the product and also applies to the context of service-oriented computing. In general, service-oriented computing emphasizes the construction of large software systems out of existing services, where services are small and self-contained pieces of software that adhere to a specified interface. Several implementations of the same interface are considered as several instances of the same service. Thereby, customers are interested in buying the best service implementation for their service composition wrt. to metrics, such as costs, energy, memory consumption, or execution time. One way to ensure the service quality is to employ *certificates*, which can come in different kinds: *Technical* certificates proving correctness can be automatically constructed by the service provider and again be automatically checked by the user. *Digital* certificates allow proof of the integrity of a product. Other certificates might be rolled out if service providers follow a good software construction principle, which is checked in annual audits. Whereas all of these certificates are handled differently in service markets, what they have in common is that they influence the buying decisions of customers. In this paper, we review state-of-the-art developments in certification with respect to service-oriented computing. We not only discuss how certificates are constructed and handled in service oriented computing but also review the effects of certificates on the market from an economic perspective.**

Keywords–Service; Certification; Service-Oriented Computing.

I. Introduction

In today's complex world, it is nearly impossible to base decisions on sufficient knowledge about all relevant facts. One way around this dilemma is to make use of certification. According to Wikipedia [1], *certification refers to the confirmation of certain characteristics of an object, person, or organization.* Real-life examples are manifold, from professional certifications, such as master's degrees awarded by universities, to product certification such as the CE conformity marking in the European Economic Area. Especially, professional certification often denotes not only procedures for validating expertise before granting a certificate, but also programs by which necessary competences are developed. Here, we restrict ourselves to the impact of the final certificates on a market. Certificates provide a certain amount of evidence that some required characteristics of a good, a person, or a process are given. No further testing is needed, except for testing the validity of the certification itself. Therefore, using widely adopted certifications is a key issue for increasing the competitiveness of a company. However, if a third party, usually a government agency, is interested in establishing minimum standards, we talk about licensing and accreditation instead of certification. Whereas, licensing is a non-voluntary process used as an entry condition of a market, accreditation is a voluntary process granting (public) recognition, e.g., for schools or universities. In this paper, we want to take a closer look at the role of certification in a service-oriented computing market.

Service-oriented computing aims at facilitating the construction of large software systems by assembling existing services. We consider services as a single, platform-independent piece of software that is specified using an interface. Service-oriented computing gives rise to a global market where numerous service providers offer their own implementations of specific services and where services of different providers interoperate with each other in *service compositions* to deliver highly customized software to users. In service compositions, a single faulty or data-leaking service may ruin the overall correctness or privacy requirements. This is a crucial characteristic of service-oriented computing as it might lead to unsatisfied customers resulting in a drop of market efficiency and finally in a market failure [2]. This can be prevented by installing quality-ensuring mechanisms that provide quality signals to customers and, hence, enable them to choose the best fitting service provider.

In online shops, it is common practice to use the reputation provided by customer reviews [3] as a quality signal. In this case, previous customers rate the delivered product or service according to their own experience. Although this quite simple example of a reputation system works well in many domains, there are some drawbacks in markets for highly specialized goods such as markets for service-oriented computing. First, since service quality is reported by users, there must be a reasonable number of people willing to provide ratings for the same service, otherwise no measure of quality exists. Because of the diversity of services and service compositions and their continuous improvement, many users will have to run services without any prior information about service quality. Second, customer ratings are subjective by nature and therefore not an objective quality signal. As service quality and performance depends on the use case, there is an additional source of unreliability. Since services are not only offered by a few well-known global players but also by unknown and untrusted entities, a reliable source for service quality information is necessary, in particular in markets for service-oriented computing.

Certification is not yet another mechanism to signal the quality of services even for inexperienced end-users. In contrast to reputation, certificates can allow judging the quality of a service or a service composition directly when it is available in the market and, thus, especially before any user might be annoyed or even harmed by a bad-quality service. In addition, the producer of a service himself is responsible for this type of trust management and bears the risk of abuse. Hence, certificates are a customer-friendly, complementary instrument besides reputation when it comes to signaling service quality. Theoretical results support this complementary interaction for certain market models [4].

Contribution. In this paper, we study certificates in service markets as a complementary approach to signal quality. On the basis of the app store example, we define all ingredients of certification in service markets that may influence the validity of a certificate. We also review state-of-the-art literature in certification wrt. to service-oriented architectures and introduce a taxonomy to classify existing approaches. Additionally, we discuss the need for certification in service markets and identify open challenges for certification in these markets.

This paper is organized as follows. In Section II, we give an overview of certification, introducing the ingredients of certification, and reviewing existing certification approaches. Section III introduces our taxonomy for certificates and classifies the reviewed certification approaches. Section IV highlights the need for certification in service markets from an economic perspective before the corresponding challenges are discussed in Section V. Finally, Section VI concludes the paper.

II. Overview of Certification

Markets in which software products are assembled and delivered on-the-fly give rise to intricacies in quality assurance. We think that these intricacies in quality assurance cannot be solved by *solely* using reputation systems and that they challenge the application of standard certification techniques. In this section, we use *app stores* as an instance of on-the-fly service markets to introduce important ingredients of certification.

App Stores as On-The-Fly Service Markets: Nowadays, smartphones often run many different apps at the same time. Often, these apps interoperate with each other. While it simplifies daily life if tasks, calendars, mails etc. are automatically kept consistent across platforms and apps, it carries a tremendous risk for the safety and privacy of user data. Typically, cross-platform consistency is achieved by storing the data in cloud storage. If such a service for cloud storage goes out of business, important data such as notes, mails and contacts might be lost forever. If these cloud services sell private user data, companies using these services might go out of business themselves. To complicate things, different service providers offer different apps realizing the same functionality – some of them might be free of charge, available only for a certain type of Android device, or might consume more or less space. Hence, users not only need to know *which* functionality they want to buy, but also need to have the chance to make informed decisions in favor or against certain providers or apps in order to find the most *reliable* provider and the app with the highest *quality* among all the existing options.

Instead of estimating the quality of a software and the reliability of a provider on the basis of user recommendations or previous experiences with other products of the same provider, nowadays, customers use certificates indicating quality such as expert recommendations or editors' choices of good apps, award winning apps, top or featured developers, apps with signed Android application packages (APKs), or trusted apps. In the following, we restrict ourselves to these examples to explain the ingredients of certification.

A. Ingredients of Certification

Essentially, *certification* is the process of attesting a specified (minimum) qualification, quality, or standard by granting a *certificate* [1] [5]. Standards to meet and guidelines for the certification process are often developed by organizations of interested parties. The certification process itself is performed by an (accredited) *certification body*. In the app store example, the roles of the standard-defining organization and the certification body issuing the certificate often coincide due to rapid advancements and missing established standards in this field. In the recommendation example, the certification body is the expert writing the recommendation, whereas the developer himself signs apps with APK keys and is thus the *certification body* in the APK key example.

The *scope of certification* identifies the entity (e.g., product, process, service) that the certification is granted for, the standard or normative document stating the *certification criteria*, and the *certification scheme* that specifies rules and procedures for testing conformity with the standard. The certifications mentioned in the app store example vary widely wrt. their scope. For instance, recommendations and signed APK keys certify single apps, whereas listings of featured developers certify app providers. As a result of a certification, a *certificate* is granted which is an accreditation that the scope of certification is met, i.e., the entity conforms to the standards wrt. the certification criteria tested. This implies that certification criteria should be objective and comparable requirements. The examples show that certificates have different *certification credibility*. Signed APK keys are digital proofs of the identity of the developer and, thereby, an objective criterion, whereas experts, which have personal preferences, may write recommendations rather subjectively. The *certificate checker* [6] [7] is the entity validating a certificate. In most cases, this will be the user. Referring to the term *trusted computing base*, we use the term *trusted base* to denote all ingredients of the certification process, which one needs to rely on to trust the certification technique. We continue with an overview of existing certification approaches (including our examples from this section) in the context of service markets.

B. Overview of Existing Certification Approaches

A broad range of certification techniques can be found in the literature and in practical use. To simplify the following discussion, we separate the presentation of the existing approaches of certificates along technical and documentary evidence. Technical evidence carries machine processable information to reinvestigate if the certification criteria are met while documentary evidence records that the certification process was performed.

1) Technical Evidence: In *Proof-Carrying Code (PCC)* [8] techniques, typically the producer himself is the certification body that carries out a formal proof of correctness wrt. some property. The certificate is made up of certain parts of the proof, which a certificate checker may use at any time to formally prove correctness wrt. the same property faster. All PCC techniques are tamper-proof, the certificate check always fails if the property is invalid and the check is correctly implemented. PCC techniques like [9] allow to fully automatically apply the PCC principle. Like the original approach [8], most of the approaches deal with functional properties. For dedicated non-functional properties, PCC approaches for software [10] and hardware services [11] exist.

Remote Attestation [12] is used to ensure the integrity of code running on a remote system, the (partial) state of a remote system, or even its behavior. To prove integrity, the remote system (the certificate body) provides collected evidence, e.g., monitoring information, information about the state or cryptographic hashes of those information. A certificate checker, e.g., the user who executes his code on a remote system, investigates if the received evidence matches its expectations.

Digital Certificates [13], which are based on cryptographic signatures, are a technique to ensure code or data integrity. A very prominent example are hash values to check the data integrity after a download. Digital certificates are also used for authentication and identification. For example, on GitHub [14], secure shell (SSH) in combination with passphrases are used for authentication, whereas Android APKs must be signed to identify the originator.

Certifying Algorithms [6] add to the result of a computation information, which witnesses the correctness of the result. Instead of certifying the result, computation certification [7] certifies the integrity of the computation. Next to the result, checkpoints such as intermediate states of the computation are returned. A certificate checker can parallelly compute for each checkpoint the subsequent checkpoint and check if the computed checkpoint is identical with the provided one.

Damiani et al. [15] propose *Web Service Certificates*. Their idea is to attach a test set for a (non-)functional property to a web service. To validate the correctness of the web service wrt. the property, the certificate checker may execute the tests. Alternatively, a third trusted authority may run the tests and provide a signed document summarizing the test result. For this alternative, Damiani et al. [16] present an approach that constructs a certificate for a service composition on the basis of the test-based certificates of all single services in the composition.

2) Documentary Evidence: To ensure reliability, Buckley et al. [17] propose pattern-based reliability certification which combines certification with monitoring. Given a reliability property plus a reliability pattern, the certification body checks if the pattern matches the property. If this is true, he adds a set of monitoring rules, each consisting of a description and a reference to a standard toolkit to monitor the corresponding metric. Thus, the monitoring rules allow to validate if the implemented service complies to the reliability pattern. A trusted certificate checker applies the monitoring rules during execution of the service to validate the reliability property. Ardagna et al. [18] combine certification and monitoring in the context of dependability certification. Based on an initial model-based prediction (a Markov model), they certify the dependability of a service for a fixed amount of time. Thereafter, an (automatic) recertification becomes necessary. When monitoring of the service execution reveals that the service currently does not fulfill the certified property, the certification body tries to downgrade the certificate, i.e., it tries to grant a certificate for a dependability policy which is derived from the original one via relaxation of some of the policy conditions. If the downgrade fails, the certificate is revoked. Note that when the real behavior again matches the originally certified property, the downgrade or revocation will be undone. The approach can also be applied to service compositions.

A common form of documentary evidence is a *Seal of Approval*. Next to the seal, a certification document is often available. The *StarAudit* [19] certificate for cloud service providers belongs to this category. To be certified, an independent certification body, an organization or an accredited auditor, performs an audit in which the offered infrastructure as a service, platform as a service, and software as a service are evaluated according to a publically available catalog of criteria. Depending on the results of the audit, either a certificate for one of three trust levels is issued for two years or no certificate is issued. The certificates are valid only if they are published on the StarAudit website. In contrast to StarAudit, the level of the *Security, Trust & Assurance Registry (STAR)* [20] certificate offered by the cloud security alliance depends on how the certification process is performed. The lowest level uses self assessment. The cloud provider must only provide a report that documents the compliance. The highest level requires continuous auditing. The specialty behind the *Certified Cloud Service* [21] seal offered by the German *Technischer Überwachungsverein (TÜV)* is that although the certificate is granted for three years, compliance is checked once a year and if compliance is no longer given the certificate is revoked. In the context of cloud services, relative documentary evidences such as a ranking of a set of cloud storage providers [22] also exist.

Participating in a certain market is sometimes also a kind of certificate if the goods or the producer must fulfill certain requirements for participation in the market. For example, in the Apple Store each app provided for download passed a review [23] that it adheres to various guidelines.

Furthermore, the quality management processes of many of today's companies are certified to be compliant with the International Organization for Standardization (ISO) 9001 standard. In the certification process, an external certification body performs an audit including interviews with employees and reviews of documents [24]. Additionally, after passing a dedicated exam, people can get a document, a certificate, that they are experts in the corresponding domain. For example, consider the Amazon web service certificates [25], which are valid for two years.

On the basis of the reviewed certification approaches, next we introduce a taxonomy for certificates.

III. Thoughts on a Taxonomy for Certificates

The goal of our taxonomy is to enable the comparison and ranking of certificates that are issued for the same scope. To that end, we identified four different characteristics.

The first criterion is the *type* of a certificate. Like in the previous section, we distinguish between two types of certificates: technical evidence and documentary evidence. Certificates that are technical evidence carry machine processable information needed by an algorithmic certificate checker to investigate whether the certification criteria are met. In

 ISBN: 978-1-61208-528-9

contrast, documentary evidence records that the certification body performed the certification process. Typical examples are seals, badges, and documents.

The second criterion is the *duration* of the validity of a certificate. For example, certificates of the technical evidence type are always valid wrt. the certified entity. Their duration is unlimited. The duration of documentary evidence may either be limited or unlimited. Often, a limited duration corresponds to a time limit [19] [20] [21] [24] [25]. However, we are aware of one approach [18] in which the duration additionally depends on the momentary status of the certified entity.

The next criterion, *quality assurance*, describes how precisely a certificate reflects the adherence of the certified entity to the certification criteria. A *dichotomous quality assurance* means that the certificate either does or does not guarantee the adherence, whereas a *gradual* quality assurance expresses that the certified entity adheres only up to a certain level to the criteria or the complete entity is not checked (e.g., only a test set is executed on the service implementation [15]). A *relative* quality assurance, e.g., a ranking like [22], ranks different certified entities and only describes that the adherence of one certified entity is better than another. If the entity that the certificate is issued for and the entity delivered to the customer are not identical, but the quality check itself is dichotomous, gradual or relative, we say that the quality assurance of the certificate for the delivered entity is *projected* dichotomous, gradual or relative.

The last criterion refers to the existence of a *countercheck*, i.e., whether it is possible for the certificate checker to check that the certified entity adheres to the certification criteria. The information carried by a technical certificate, e.g., a mathematical proof [8], a cryptographic hash [13], monitored data [12], or a test suite [15], naturally imposes such a countercheck. Counterchecks for certificates based on documentary evidence include monitoring [17] [18] or sample examination. Table I gives an overview of the approaches discussed in Section II-B and their classification.

We continue with a detailed motivation for certification in service markets.

IV. On Service Quality in Service Markets

One underlying characteristic of service markets is that heterogeneous products, i.e., different implementations of the same service but, for example, with diverse non-functional properties are offered. Without further information about a service implementation, service markets, as well as any market for experience goods, face the problem that before the purchase producers, the developers, in the case of service markets, and customers have different levels of knowledge about the service implementation's functional and especially non-functional properties. This constellation is called information asymmetry and arises whenever a product or service is traded whose full characteristics are revealed to the customer only after he bought and experienced it.

Information asymmetry is one reason why high-quality products are driven out of the market in market constellations *without signals* such as customer reviews, which reflect the product quality [2]. The reason is that developing a service implementation with better (non-)functional properties, e.g., a better performance, is typically more expensive. High-quality developers set higher prices to cover their expenses. However, without the revelation of the service quality low-quality developers may also set the same price to increase their profits. Customers who are aware of different quality levels do not trust the developers and show a lower willingness to pay these higher prices. Hence, the prices decrease, high-quality developers cannot cover their expenses and they go bankrupt. In contrast, low-quality developers can cover their expenses with low market prices and remain in the market whereby high quality is finally driven out of the market.

In addition, online service markets have to deal with two further characteristics. First, there is a high fluctuation of service providers, i.e., there are numerous developers which enter and leave the market at any time. Second, often various services are composed to sell a complete software solution.

These characteristics may cause problems even in service markets with an *existing reputation system* especially when new developers enter the market. First, the developers face the moral hazard problem since they can deceive their customers without fearing any sanctions. They can claim to sell an efficiently working service with an appropriate price although their services are working inefficiently. Obliging customers, who buy these services because they are cheap, will be unsatisfied and leave the market for future purchases of services due to this so called adverse selection [26]. Second, the developers indeed offer a high-quality service, but because they have no positive reputation, potential buyers are not willing to pay the demanded price. Hence, the suppliers have two options: 1) They can reduce their prices and invest in a reputation or 2) they can leave the market. Neither of these alternatives is desirable since one party is unsatisfied. In addition, when single services are combined into software packages, customers are hardly enabled to review the single components. Hence, reviews are written for the whole composition and it is difficult to establish reputation for single services since these reviews must be disaggregated to assess the single services.

To overcome these likely occurring problems, information signals on quality can be induced into the market. There are two ways to provide this information: signaling and monitoring. *Monitoring* is the most often used approach in online markets. Customers who already have experienced the seller's service write a review and in this way monitor the observed quality. The purpose of this approach is to build up trust [27]. As already stated above, this approach comes along with problems, in particular for new market participants. This is a severe disadvantage especially in markets for service compositions without few 'big players' but with numerous small and specialized service developers.

The second approach is for the seller to *signal* the quality by showing his trustworthiness. This can be implemented by offering warranties, presenting satisfaction guarantees, or testing the product by a third party and receiving a certificate in return [28]. In markets for service compositions, the signaling approach has weighty advantages: 1) New market participants can reveal the true quality of their services from the beginning. 2) The costs for signals are borne individually by the service developers. 3) The necessity of the disaggregation of customer ratings is removed: The above-mentioned problem of disaggregating customer reviews to assess the single services is bypassed when the quality is signaled by every single developer.

 ISBN: 978-1-61208-528-9

Table I. CLASSIFICATION OF EXISTING CERTIFICATION APPROACHES ACCORDING TO OUR PRELIMINARY TAXONOMY

Approach	Type	Unlimited Duration	Quality Assurance	Countercheck
Amazon Web Service Certificate [25]	documentary	×	gradual	×
Apple Store App [23]	documentary	✓	gradual	×
Certified Cloud Service [21]	documentary	×	gradual	×
Certifying Algorithm [6]	technical	✓	dichotomous	✓
Computation Certification [7]	technical	✓	dichotomous	✓
Dependability Certification [18]	documentary	×	gradual	✓
Digital Certificate [13]	technical	✓	dichotomous	✓
ISO 9001 [24]	documentary	×	gradual	×
Pattern-Based Reliability Certification [17]	documentary	✓	gradual	✓
Proof-Carrying Code [8]	technical	✓	dichotomous	✓
Ranking	documentary	?	relative	×
Remote Attestation [12]	technical	✓	dichotomous	✓
STAR [20]	documentary	×	gradual	×
StarAudit [19]	documentary	×	gradual	×
Top Developer Award	documentary	?	gradual	×
WS-Certificate [15]	technical	✓	gradual	✓

These advantages show the potential of certificates in service markets although they come along with the challenges we address in the next section.

V. Challenges for Certification in Service Markets

On-the-fly markets challenge the use of standard certification techniques. For instance, offered services, the market itself, and also market participants are heterogeneous. Thus, services with technical certificates compete or even interoperate with services provided by companies with a certified workflow. Also, solutions to a user request, the (composed) service plus the execution environment, are created on-demand and hence challenge the creation of certificates in time. Addressing these and further issues in this chapter, we show future directions of research on certification in on-the-fly markets.

A. Composition of Certificates

At the core of on-the-fly computing is the configuration of service compositions out of existing services. Certification must be become *compositional*. First, it is important to find out which types of certificate criteria are compositional at all, e.g., expected runtime might not be compositional if services generate unusual data with a high probability. Second, certification processes for compositional properties must be defined. This is especially interesting for the certification of functional properties. Important questions to answer include whether service compositions can be certified only if all services used have a special technical certificate and how to define technical certificates for models of service compositions. Considering the limited duration of certificates, procedures have to be defined to deal with a composition of certificates when for a subset of services the certificates are expired. This is even more important when the expiration is not necessarily a consequence of time-limited validity but also of varying service quality. A third issue addresses the customers' perception of certificates from different sources. If customers trust different certificates (e.g., those with technical and documentary evidence) differently this must be taken into account since certificates for compositions might not lead to trust though most of the services are certified with trust-building certificates.

B. On-the-Fly Certification

In on-the-fly markets, user requests are not known in advance and often a request must be served, which has not been entered into the market before. In this case, a new solution must be created for the user request. Simultaneously, the newly created solutions must be certified. Typically, issuing a certificate is laborious, e.g., resource and time consuming. The high costs, especially a high issuing time, conflicts with an on-the-fly offer of a solution, i.e., a user gets an offer after a few seconds of his request. Thus, one must rethink the certification process to reduce the certification effort of the new solution, a (composed) service plus the execution environment. An important question is which tasks of the certification process can be done offline in advance. For example, to certify worst case execution time, the worst case execution time of single services can be computed in advance for the available processor architectures in the market. To certify the new solution, one can treat the single services in the composition as a black box considering the worst-case execution time corresponding to the processor architecture that the service will run on. Thus, one only needs to analyze the paths of the composition.

C. Business Secrets

Certificates disclose attributes of the underlying service, which may conflict with business secrets. Composing multiple certified services scales up this problem since two sources of information are supplied with each service. A certified composition of services likely reveals information about the composition process performed by a so called on-the-fly provider and about the single services constructed by several service providers. More importantly, service providers must grant on-the-fly providers, potential competitors in the market, sufficient insights into their single services for certification of the composed service. Moreover, certificates are not checked by the customer himself, but by third entities such as compute centers or on-the-fly providers.

D. Economic Perspective

Certification is tied to costs and benefits in consequence of the reduction of information asymmetries and the customers' higher willingness to pay. In conventional market situations this value is easy to evaluate by comparing certified products with non-certified ones. Hence, suppliers will bear the costs of certification if benefits exceed them. In markets for composed services with compositions including different types of certificates, this suppliers' trade-off is much harder to handle since benefits can hardly be estimated. Mechanisms must be found to predict suppliers' benefits and to distribute them among

 ISBN: 978-1-61208-528-9

the single suppliers. This is even more difficult as different certificates with different signal power and costs are employed.

E. Realization of Certification

To integrate certification in distributed service markets such as on-the-fly markets, one must make several design decisions. First, one must choose how to realize the certification bodies. A certification body could be just a service offered by the market infrastructure or it could become a profit-oriented market participant. In the latter case, one must deal with strategic behavior of the certification bodies. For example, they may be corrupt and grant a certificate although the certified entity does not meet the certification criteria, or they may offer certificates ensuring insignificant properties. Second, it must be specified who defines the scope of certification, i.e., the certification criteria, the certification scheme, and so on. For example, one could use a consortium of the market participants or certification bodies can decide themselves. These decisions are important since they have strong and direct implications on the whole market. Other design decisions relate to contracts, penalties for misuse of certificates, or dealing with customer complaints regarding certificates.

F. Certification Impact on Service Markets

So far, certificates are perceived as add-ons to single services. It is unclear whether a *certification system* in the sense of a centralized reputation system (as described in [29]) has implicit consequences on the market functionality. The quality of offered services might be affected and therefore also the market prices. The dynamics of these effects on entities, services, compositions, or participants are unclear. In addition, certificates can be considered as entry requirements on some markets, e.g., the Apple app store, as every app is checked by Apple before it is listed in the store, whereas an app does not need to have any certificate to be published in Google's Play Store. Hence, a high weight of certificates, such as an entrance requirement in the Apple store, might also distinguish different markets (not only apps or services) from each other. Further, interactions of certification systems, e.g., with present reputation systems need to be investigated. It is not certain whether these systems address different dimensions of service attributes or not and, hence, whether these systems are complements or substitutes.

VI. CONCLUSION

We presented the necessary ingredients for certification, reviewed existing certification approaches, and classified them according to our taxonomy. Additionally, we motivated the need for certification in service markets and discussed open problems wrt. their usage.

ACKNOWLEDGEMENTS

This work was partially supported by the German Research Foundation (DFG) within the Collaborative Research Centre "On-The-Fly Computing" (SFB 901).

REFERENCES

[1] Wikipedia, The Free Encyclopedia, "Certification," 2017, https://en.wikipedia.org/wiki/Certification [retrieved: 01-10-2017].

[2] G. A. Akerlof, "The market for "lemons": Quality uncertainty and the market mechanism," The Quarterly Journal of Economics, vol. 84, no. 3, pp. 488–500, 1970.

[3] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman, "Reputation systems," Commun. ACM, vol. 43, no. 12, pp. 45–48, 2000.

[4] M. van der Schaar and S. Z. Zhang, "A dynamic model of certification and reputation," in EC '14, pp. 967–968, ACM, 2014.

[5] ISO, "Certification," 2017, http://www.iso.org/iso/home/standards/certification.htm [retrieved: 01-10-2017].

[6] R. McConnell, K. Mehlhorn, S. Nher, and P. Schweitzer, "Certifying algorithms," Computer Science Review, vol. 5, pp. 119–161, 2011.

[7] S. M. Khan and K. W. Hamlen, "Computation certification as a service in the cloud," in CCGrid '13, pp. 434–441, IEEE, 2013.

[8] G. C. Necula, "Proof-carrying code," in POPL '97, pp. 106–119, ACM, 1997.

[9] M.-C. Jakobs and H. Wehrheim, "Certification for configurable program analysis," in SPIN '14, pp. 30–39, ACM, 2014.

[10] K. Crary and S. Weirich, "Resource bound certification," in POPL '00, pp. 184–198, ACM, 2000.

[11] T. Wiersema and M. Platzner, "Verifying worst-case completion times for reconfigurable hardware modules using proof-carrying hardware," in ReCoSoC '16, pp. 1–8, June 2016.

[12] G. Coker, J. Guttman, P. Loscocco, A. Herzog, J. Millen, B. O'Hanlon, J. Ramsdell, A. Segall, J. Sheehy, and B. Sniffen, "Principles of remote attestation," Int. J. Inf. Secur., vol. 10, no. 2, pp. 63–81, 2011.

[13] W. Diffie and M. Hellman, "New directions in cryptography," IEEE Transactions on Information Theory, vol. 22, no. 6, pp. 644–654, 1976.

[14] "Ssh passphrase on github," 2017, https://help.github.com/articles/working-with-ssh-key-passphrases/ [retrieved: 01-10-2017].

[15] E. Damiani, N. E. Ioini, A. Sillitti, and G. Succi, "Ws-certificate," in 2009 Congress on Services - I, pp. 637–644, IEEE, 2009.

[16] M. Anisetti, C. A. Ardagna, and E. Damiani, "Security certification of composite services: A test-based approach," in ICWS, pp. 475–482, IEEE, 2013.

[17] I. Buckley, E. B. Fernandez, M. Anisetti, C. A. Ardagna, M. Sadjadi, and E. Damiani, "Towards pattern-based reliability certification of services," in OTM '11, pp. 560–576, Springer, 2011.

[18] C. A. Ardagna, R. Jhawar, and V. Piuri, "Dependability certification of services: a model-based approach," Computing, vol. 97, no. 1, pp. 51–78, 2015.

[19] EuroCloud, "Staraudit," 2017, https://staraudit.org/ [retrieved: 01-10-2017].

[20] C. S. Alliance, "Security, trust & assurance registry (STAR)," 2017, https://cloudsecurityalliance.org/star/ [retrieved: 01-10-2017].

[21] TÜV, "Certified cloud service," 2017, http://www.tuv.com/cloud/ [retrieved: 01-10-2017].

[22] PCWELT, "Ranking cloud storage," 2017, http://www.pcwelt.de/ratgeber/Onlinespeicher-ohne-NSA-Die-besten-kostenlosen-Cloudspeicher-in-Deutschland-Schweiz-9829982.html [retrieved: 01-10-2017].

[23] A. Inc., "App review," 2017, https://developer.apple.com/app-store/review/guidelines/ [retrieved: 01-10-2017].

[24] ISO, "ISO 9000—selection and use," 2009, ISO Technical Committee ISO/TC 176.

[25] Amazon, "Amazon webservice certification," 2017, https://aws.amazon.com/certification/ [retrieved: 01-10-2017].

[26] G. Lewis, "Asymmetric information, adverse selection and online disclosure: The case of ebay motors," The American Economic Review, vol. 101, no. 4, pp. 1535–1546, 2011.

[27] P. Resnick and R. Zeckhauser, "Trust among strangers in internet transactions: Empirical analysis of ebays reputation system," The Economics of the Internet and E-commerce, vol. 11, no. 2, pp. 23–25, 2002.

[28] D. Dranove and G. Z. Jin, "Quality disclosure and certification: Theory and practice," J. Economic Literature, vol. 48, no. 4, pp. 935–963, 2010.

[29] S. Brangewitz, A. Jungmann, R. Petrlic, and M. Platenius, "Towards a flexible and privacy-preserving reputation system for markets of composed services," in SERVICE COMPUTATION '14, pp. 49–57, IEEE, 2014.

Security Considerations Based on Classification of IoT Device Capabilities

Seungyong Yoon, Jeongnyeo Kim, Yongsung Jeon
Mobile Security Research Section
Electronics and Telecommunications Research Institute
Daejeon, Rep. of Korea
e-mail: syyoon@etri.re.kr, jnkim@etri.re.kr, ysjeon@etri.re.kr

***Abstract*— In the Internet of Things (IoT) environment, the various types of IoT devices, from tiny and lightweight devices to powerful smart devices, are connected to send and receive information to and from each other. We need to define different security requirements of IoT devices, depending on the functionality, capabilities, and characteristics. In this paper, we analyze security threats and vulnerabilities of IoT devices, and propose the security requirements based on classification of IoT device capabilities.**

Keywords-IoT; security; classification.

I. INTRODUCTION

With the rapid growth of IoT market, security threats and vulnerabilities of IoT devices, having a variety of types and performance, are increased by being interconnected with the network. Therefore, IoT devices need to be provided with various security features for responding to security threats such as hacking and exploitation. Most existing IoT devices are available for the CPU and memory resources are limited, such as lightweight devices, so low-power and lightweight security mechanism are essential. In addition, IoT devices should be able to prevent malfunction or stop due to the malicious code, and should be able to prevent information leakage from physical theft or loss.

In this paper, we analyze security threats and vulnerabilities of various IoT devices, and propose security requirements in order to prevent confidentiality, integrity and availability, for example, access control, device authentication, data integrity, anti-virus, secure storage, security updates, and so on. Also, we classify IoT devices into four categories depending on their capabilities and propose security requirements for each class of IoT devices.

The rest of this paper is organized as follows. Section II gives an overview of related work and provides a discussion of our contribution. Section III describes security threats analysis and security requirements of IoT devices. Section IV proposes security requirements based on classification of IoT device capabilities, followed by conclusion in Section V.

II. RELATED WORK

IoT devices can be classified according to various criteria. They can be classified by functionality and type [1], classified based on the type of data handled [2], and classified according to the degree of resource constraint [3]. Many researches have been performed for the security function required of such IoT devices [4][5][6] and classification mechanism for IoT devices to determine their capability to support security mechanisms of different degrees [7].

III. SECURITY THREATS ANALSYSIS AND SECURITY REQUIREMENTS OF IOT DEVICES

IoT devices are exposed to a variety of security threats and vulnerabilities. Hackers who launch attacks using these vulnerabilities exhibit malicious behaviors. Typical security threats and vulnerabilities of IoT devices include unauthorized access, loss or theft, physical destruction, information leakage, illegal data modification, and denial of service attacks.

TABLE I. SECURITY THREATS AND VULNERABILITIES OF IOT DEVICES

Categories	Security threats	Security vulnerabilities
Confidentiality	Eavesdropping, Man-in-the-middle attack, Illegal message modification, Sniffing	Sensitive data(privacy) leak
	Zombie devices, Distributed Denial of Service (DDoS) attacks, Phishing, Pharming	User data leak, Secondary damage caused by malware infection
	Attacks using web interface vulnerabilities	Data and device takeover
	Illegal firmware update, Hardware interface and flash memory physical takeover	Root privilege takeover
	Replication through the device unique identifying information leakage and change	Data and device replication
Integrity	Eavesdropping, Man-in-the-middle attack, Illegal message modification	Sensitive data(privacy) leak
	Attacks of interfaces and system vulnerabilities through illegal intrusion and access	Firmware and operating system permissions takeover
Availability	Software malware infection	malfunction
	Physical removal and destruction, Unusual installation attempt	Data and device takeover
	Persistent attempted access attack, Denial of service attack	System operation and malfunction
	Lost or stolen, installation and disposal	Data and device takeover

	Network platform smishing attack	Malware infection
	Network platform screen capture attack	Sustainable security error
Authentication/ Authorization	Unauthorized user access and unauthorized devices access	Privilege takeover, Illegal access, Data leak
	Device replication, alteration, appropriation	Data and device takeover

The openness of IoT platform accelerates interworking between heterogeneous devices, and the variety of security threats is increasing. In addition, three elements of information security which consist of confidentiality, integrity, and availability are increasing the possibility of infringement. From these threats, we present the security requirements to keep the IoT devices safe.

A. *Confidentiality*

- [Transmitted message encryption] Messages transmitted between IoT devices are to be transmitted in encrypted format to prevent illegal sniffing or eavesdropping.
- [Malware response] IoT devices should provide the ability to detect and defend against malware infections and external hacker attacks, such as worms and viruses to prevent information leakage.
- [Data encryption] IoT devices should encrypt sensitive data such as private information and cryptographic key, and securely process and store these data to prevent information leakage.
- [Tamper resistance] IoT devices should provide tamper resistance function to ensure the safety and reliability from physical attacks.
- [Device ID management] IoT device should have unique device identification information and safely handled so as not to leak outside or to change illegally.

B. *Integrity*

- [Data integrity] IoT device should provide data integrity verification function to prevent forgery of data.
- [Platform integrity] IoT devices should provide platform integrity verification function of system-level such as firmware and operating system.
- [Secure booting] When power is first introduced to the device, IoT devices should provide secure booting function to ensure the reliability of the device through authenticity and integrity of the software on the device.

C. *Availability*

- [Logging] IoT device should provide the appropriate log function for the user, the system, the security event.
- [State Information Transmission] IoT device should provide a periodic keep-alive message or device state information transmission function for prevention from physical removal/destruction and abnormal installation attempt.
- [External attack response] IoT device should provide the capability to respond to external attacks, such as denial of service attacks and persistent connection attempt attack.
- [Security monitoring/management] IoT devices should provide security monitoring and management capabilities to respond adequately if lost or stolen, installation and disposal, etc.
- [Security patch] IoT device should provide a safe and secure software update and patch function.
- [Security policy setting] IoT device should provide the capability to securely set an appropriate security policy on the various types of devices.
- [Software safety] IoT devices should ensure software safety, with features such as appropriate module separation or removal, and access restrictions, despite a software failure or malfunction due to malware infections.

D. *Authentication/Authorization*

- [User authentication] IoT device should provide a user authentication function to block the access of unauthorized users.
- [Device authentication] IoT device should provide a device authentication function in order to block the access of illegal device.
- [Password management] IoT device sets the secure and robust password, and should provide the periodic update feature.
- [Mutual authentication] IoT device should provide a mutual authentication between the devices to establish secure, autonomous communication environment.
- [Authority control] IoT device should provide the authority control functions, such as ownership control for preventing information leakage and privacy protection.
- [Access control] IoT device should provide a access control function to block the access of unauthorized users and devices.
- [Identification information verification] IoT device should provide the unique device identification information verification function for preventing device replication, alteration, and appropriation.

IV. SECURITY REQUIREMENTS BASED ON CLASSIFICATION OF IOT DEVICE CAPABILITIES

International Telecommunication Union (ITU) classified into four different types of IoT devices according to type and functionality as follows: data-carrying device, data-capturing device, sensing and actuating device, and general device [1]. In this paper, however, we classify IoT devices in four classes, depending on their capabilities.

Class 0 devices are very constrained devices, such as compact, lightweight, and low-power sensors. Due to the constrained in memory and processing capability, they do not participate in Internet communication in a secure way. These devices usually communicate with the help of proxies or gateways.

Because of constrained resource and processing capabilities, Class 1 devices cannot easily communicate with other devices employing a full protocol stack, such as HyperText Transfer Protocol (HTTP) and Transport Layer Security (TLS). They use a protocol stack for specifically designed for IoT device with constraints, such as Constrained Application Protocol (CoAP). Device examples include a blood glucose meter or a thermostat that is based on 8-bit or 16-bit processors. It is possible to communicate with other devices without the help of a gateway.

TABLE II. SECURITY REQUIREMENTS ACCORDING TO IOT DEVICES CAPABILITIES

Categories	Security Requirements	Class 0	Class 1	Class 2	Class 3
Confidentiality	Message encryption		√	√	√
	Malware response				√
	Data encryption		√	√	√
	Tamper resistance			√	√
	Device ID management	√	√	√	√
Integrity	Data integrity		√	√	√
	Platform integrity			√	√
	Secure booting			√	√
Availability	Logging			√	√
	State Info. Transmission	√	√	√	√
	External attack response				√
	Security monitoring			√	√
	Security patch			√	√
	Security policy			√	√
	Software safety		√	√	√
Authentication/ Authorization	User authentication		√	√	√
	Device authentication		√	√	√
	Password management		√	√	√
	Access control		√	√	√
	Device ID verification			√	√

Class 2 devices can be supported in the existing communication protocol stack, or that are less constrained. Examples include an IP camera or a smart meter that is based on 32-bit processors. However, these devices also can benefit from using low-power and lightweight protocol, and from consuming less bandwidth.

Class 3 device example is a smartphone or a tablet beyond class 2. They can use existing protocols without any changes or modifications. However, these devices can still be constrained by a limited power supply.

Table II presents the security requirements according to IoT device classification. It can be easily utilized as a security guideline to apply to various IoT devices.

V. CONCLUSION

IoT devices are always exposed to security threats, such as loss or theft, information leakage, and data forgery. In this paper, we analyzed security threats and vulnerabilities for IoT devices, and proposed security requirements based on classification of IoT device capabilities. By presenting with the applicable security requirements in the various classes of IoT devices, we are expected to contribute to improving security of IoT devices.

ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (B0190-16-2032, Development of Operating System Security Core Technology for the Smart Lightweight IoT Devices).

REFERENCES

[1] ITU-T Y.2060, Overview of the Internet of Things, 2012.6.

[2] Imagination Technologies Limited, White Paper, Internet of Things – Opportunities for device differentiation, 2015.2.

[3] IETF RFC7228, Terminology for Constrained-Node Networks, 2015.4.

[4] Wind River, White Paper, Security in the Internet of Things, 2015.1.

[5] R. Roman, P. Najera, and J. Lopez, “Securing the Internet of Things”, IEEE Computer, Vol.44, pp.51-58, 2011

[6] S. Babar, P. Mahalle, A. Stango, N. Prasad, and R. Prasad, “Proposed Security Model and Threat Taxonomy for the Internet of Things (IoT)”, Proc. International Conference on Network Security and Applications (CNSA 2010), Springer-Verlag Berlin Heidelberg, pp. 420-429, 2010

[7] V. J. Jincy and S. Sundararajan, “Classification Mechanism for IoT Devices towards Creating a Security Framework”, In Intelligent Distributed Computing, Springer International Publishing, pp. 265-277, 2015

 ISBN: 978-1-61208-528-9

Agent based Framework for QoS Measurement applied in SOA

A uniform Approach based on a QoS Meta Model

Andreas Hausotter, Arne Koschel
University of Applied Sciences & Arts Hannover
Faculty IV, Department of Computer Science,
Hannover, Germany
email: Andreas.Hausotter@hs-hannover.de
email: Arne.Koschel@hs-hannover.de

Johannes Busch, Markus Petzsch, Malte Zuch
University of Applied Sciences & Arts Hannover
Faculty IV, Department of Computer Science,
Hannover, Germany
email: Johannes.Busch@stud.hs-hannover.de
email: Markus.Petzsch@stud.hs-hannover.de
email: Malte.Zuch@hs-hannover.de

***Abstract*—Nowadays, enterprises are faced with a variety of major challenges, such as the cut-throat competition in a global market, a decreasing customer loyalty, and the strategic adjustment moving from a product centric perspective to a customer centric perspective. Therefore, businesses need to change their operational processes in a flexible and agile manner to keep their competitive edge. A Service-oriented Architecture (SOA) may help to meet these needs. As the application landscape of enterprises is inherently heterogeneous and highly distributed it is a great challenge to provide services with a certain quality. This is particularly the case when services are requested externally via the web. Therefore, quality of service (QoS) measurement and analysis is a crucial issue in Service-oriented Architectures. As the key contribution of this paper we present a generic SOA Quality Model (SOA QM) based on the measurement standard ISO/IEC 15939, a SOA Information Model (SOA IM), and an architectural concept of a QoS System. The SOA IM is an XML-based specification for the measurement to be performed. The QoS System provides an execution platform for the SOA IM, based on a Complex Event Processing (CEP) approach and guarantees minimal impact on the SOA environment. The concepts are explained in detail using a standard process of the German insurance domain.**

Keywords—Service-oriented Architecture (SOA); Quality of Service (QoS); Measurement Process; Complex Event Processing (CEP).

I. Introduction

Distributed IT-systems are commonly used in today's companies to fulfill the needs of agility and scalability of their business processes to manage the highly variable demand of the market. Typical scenarios are real time logistics and delivery, just in time supply chain management and in general, handling services in real time to fit market demands.

The latter is commonly used within the finance and insurance industry during their internal computation of risk and money management and for their external customer services, like proposal calculations (including the current market conditions). Especially the external services must have a high quality in terms of time behavior. Google has shown that a latency of 100 ms up to 400 ms causes an impact of -0.2 % up to -0.6 % concerning the daily usage of web services by the customers [1]. The integration of those services to run business processes in a stable way, fulfilling the varying demand of the market, is commonly realized with Service-oriented Architectures (SOA).

Those architectures integrate (micro)-services within distributed systems to run business processes with a high capability in terms of agility. Especially the distribution of the services over several systems allows to scale with the market demands.

Distributing and handling several services is a common concern of the insurance industry. But an increasing distribution and more complex business processes will only gain more agility with SOA, if the distribution of the services over several systems is realized in a reasonable way. For getting the required control of the distribution of those services, a measurement system is required. Measuring the general QoS in distributed systems is part of the motivation of this work and is explained in detail in the next subsection. The subsection after the discussion of the general motivation will show the contribution to the general problem in measuring the QoS in SOA within the application scenario of the insurance industry, explained in section III.

A. Motivation

In many cases, it is not foreseeable to forecast, how much computing power and bandwidth the infrastructure needs to host the allocated services within the distributed computing system. Beside these design decisions of the infrastructure, there is a further problem in allocating the services to the right locations within the distributed system. This allocation will influence how much bandwidth and calculation power is available for the services and how many services will share identical resources during the same time. So if several services will use the same part of the infrastructure, this could lead to increasing latencies over the whole system, resulting in an unfavorable time behavior for the users. Especially if some services are requested with intense demands of the market, latencies could rise in an unpredictable manner.

Such a scenario is typical for the German insurance industry. At the end of the year, millions of users are able to switch their insurance contracts and will request therefore designated online services. The general demand is not foreseeable and the intensely interaction between the insurance industry and

 ISBN: 978-1-61208-528-9

the finance industry requires a high quality of those services. Especially the historically low interest rates in today's market provokes fast changing business models and the need of a fast adoption to new business processes and the ability to offer services in a high quality to fulfill the external user demands and the internal interaction within the finance industry. To fulfill those demands, distributed systems with SOA will benefit from an across boarder measurement of the quality of those services, especially in terms of latency. Such a measurement system is the contribution of this work and is explained in the following subsection.

B. Contribution

The need for a new development of a flexible measurement system is influenced by the limitations of common solutions. The scenario of this work is based on a German insurance company, which already uses Dynatrace as a measurement solution [2].

The partner from the insurance industry is currently restructuring and modernizing his business processes and therefore, he needs a more flexible and generic approach to integrate an external measurement system for monitoring and analyzing the time behavior of his services. Additionally, a more detailed analyzer component was required to process the measured data.

So on the one hand, the approach has to be integrated in a generic way with minimal interaction points within the SOA of the partner from the insurance industry to guarantee a simple integration during the continuous development process. But on the other hand, the solution should offer a flexible and detailed analyzer component.

This article will present our currently ongoing applied research work. Since it is still "work in progress", we will mostly focus on measurement concepts and an adequate measurement model here. We combine this with an initial description of the main insurance application scenario used by us. More technical details on our actual prototypic implemenation as well as QoS measurement results, will be presented in future work.

The required solution was defined by the following:

- generic approach to generate the measurement system,
- automatic integration of the measurement system in the existing SOA,
- lose couplings within the existing SOA,
- flexible agent based approach,
- technology independent approach using standards (XML),
- individual and customizable analyzer component.

As stated above, beside technical concepts we will also present some details from our mainly utilized application scenario, which is based upon the ideas from the "Check 24" process. Within this process different offerings for the same kind of insurance are compared. The offerings typically origin from several insurance companies. They are, for example, different offerings for car insurances. Based on certain input parameters, the end user gets eventually different insurance offers by this process. The proposal service used by "Check 24" is a common service throughout the German insurance sector is implemented by various insurance companies.

This service can be called externally by applications such as "Check 24" through a common interface given by a so called "BiPro specification". BiPro is widely used throughout the German insurance sector and the availiabilty of these services has a significant impact on competitiveness. Internally the proposal service is, for example, used in the process "Angebot erstellen" ("create proposal") of the general German "Versicherungsanwendungsarchitektur (VAA)" (cf. [32]), which describes a set of standardized insurance processes working within a generalized "insurance application architecture". Our project partner has implemented a similar process for it's own agent respectively customer portal.

The remainder of this paper is structured as follows: In the next Section II we discuss some related work. Section III describes our application scenario in some detail. In Section IV and Section V our general Quality of Service (QoS) measurement model and concept are described. Eventually Section VI concludes this paper and gives some outlook to future work.

II. PRIOR AND RELATED WORK

In prior work, we already discussed several aspects of the combination of SOA, Business Process Management (BPM), Workflow Management Systems (WfMS), Business Rules Management (BRM), and Business Activity Monitoring (BAM) [16][17][15] as well as Distributed Event Monitoring and Distributed Event-Condition-Action (ECA) rule processing [20][21]. Building on this experience, we now address the area of QoS measurement for combined BRM, BPM, and SOA environments within the (German) insurance domain context.

Work related to our research falls into several categories. We will discuss those categories in turn.

General work on (event) monitoring has a long history (cf. [12][13] or the ACM DEBS conference series for overviews). Monitoring techniques in such (distributed) event based systems are well understood, thus such work can well contribute general monitoring principles to the work presented here. This includes also commercial solutions, such as the Dynatrace [2] system or open source monitoring software like, for example, the NAGIOS [14] solution. In those systems there is however, generally not a focus on QoS measurement within SOAs. Also, they usually do not take application domain specific requirements into account (as we do with the insurance domain).

Active DBMS (ADBMS) offer some elements for use in our work (see [18][19] for overviews). Event monitoring techniques in ADBMSs are partially useful, but concentrate mostly on monitoring ADBMS internal events, and tend to neglect external and heterogeneous event sources. A major contribution of ADBMSs is their very well defined and proven semantics for definition and execution of Event-Condition-Action (ECA) rules. This leads to general classifications for parameters and options in ADBMS core functionality [19]. We may capture options that are relevant to event monitoring within parts of our general event model. QoS aspects are handled within ADBMS, for example, within the context of database transactions. However, since ADBMSs mostly do not concentrate on heterogeneity (and distribution), let alone SOAs, our work extends research into such directions.

The closest relationship to our research has work, which directly combines the aspects QoS and SOA. Since about 2002 several articles fall into this category. However, in almost all known articles the SOA part focuses on WS-* technologies. This is in contrast to our work, which takes the operational environment of our insurance industry partners into account.

Examples of WS-* related QoS work include QoS-based dynamic service bind [26][27], related WS-* standards such as WS-Policy [22], and general research questions for QoS in SOA environments [23].

Design aspects and models for QoS and SOA are, for example, addressed in [28][24][33][25][26], SOA performance including QoS in [34], and monitoring for SOA is discussed in articles such as [30][31][29][35].

III. APPLICATION SCENARIO

Customers are using online platforms to compare the conditions and proposals offered by different companies. The online platform check24.com allows customers to compare different insurance proposals. Therefore, the insurance companies need to respond to those requests to be aware for potential customers on such platforms. The underlying scenario for this work is a service for calculating individual proposals for such online platforms. This scenario is automatically requested by the online customer information platform and needs to respond in a timely manner. The business process for calculating the proposal follows four steps:

- check input parameters for plausibility,
- call all additional relevant services to get required data,
- calculate the proposal based on internal business rules,
- deliver the proposal to the requesting online platform.

The partner from the insurance industry has already developed a distributed system to create and run such business processes. This system uses the approach of SOA and integrates various micro-services located across several locations.

Measuring the time behavior is a feasible approach to maintain the overall system and scale it to changing market demands to fulfill the required quality of such services (QoS). The distributed system is designed with the concept illustrated in Fig. 1. The system part alpha is the enterprise service bus (ESB) of the system, which is responsible to integrate the business processes with further applications and services. Those business processes are parameterized by specific business rules, stored in a business rule database.

The communication with this business rule database is realized via web service calls. So in general, alpha is the central communication component of the system.

The system part beta is the current process engine to run the business processes and is connected via JMS with alpha. Those business processes are influenced by the stored business rules and the business process data, which are stored in a separated database.

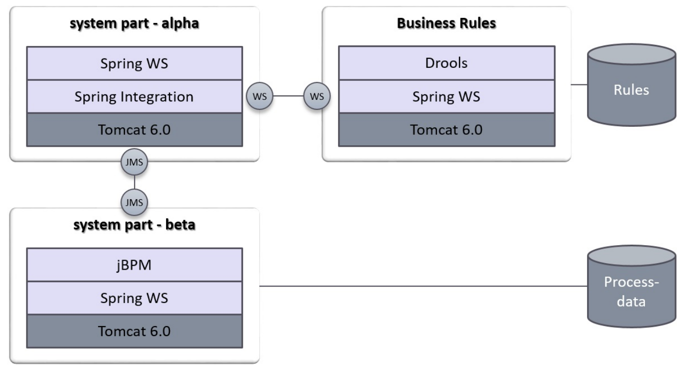

Figure 1: The application scenario

This distributed system defines the scenario where several services are parameterized, called and integrated (via alpha) over several locations. The generic and XML-based measurement concept of this work will use this scenario to measure QoS-Parameters, especially the time behavior of services. The specific measurement model is described in the next section.

IV. MEASUREMENT MODEL

The assessment of the QoS in Service-oriented Architectures is based on a *SOA Quality Model (SOA QM)*, which combines characteristics and sub-characteristics in a multilevel hierarchy. For this purpose we adjusted the ISO/IEC-Standard 9126 to meet the SOA-specific requirements. Fig. 2 illustrates the characteristics, sub-characteristics and relationships between these concepts. In our research work we will focus on *Time Behavior,* which contributes to *Efficiency*.

Although ISO/IEC 9126 was revised by the ISO/IEC-Standard 25010 (*Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models*, cf. [9][6]) we use ISO/IEC 9126 as a starting point because of it's high degree of awareness in German-speaking countries (cf. [7]). Moreover, the German version of the ISO/IEC 25000 series has been prepared by the German Institute for Standardization (DIN) but is not yet available (cf. [11])

Instead of applying the quality metrics devision of SQuaRE (i.e. ISO/IEC 2503x), our approach is based on the comprehensive ISO/IEC-Standard 15939 (cf. [9]). The basic model, as to be found in similar form in the contribution of Garcia et al. ([10]) has been aligned and extended by quality requirements, quality models, and some system components. In the following subsections we describe the main concepts of our *SOA Measurement Information Model (SOA MM)* as shown in Fig. 4.

A. Information Need and Information Product

The determination of the QoS in a SOA is always demand-driven, since both the specification ('*What* and *how* should be measured?') as well as the execution of the measurement itself and the subsequent interpretation of the results can cause a significant organizational and technical effort.

Copyright (c) IARIA, 2017. ISBN: 978-1-61208-528-9

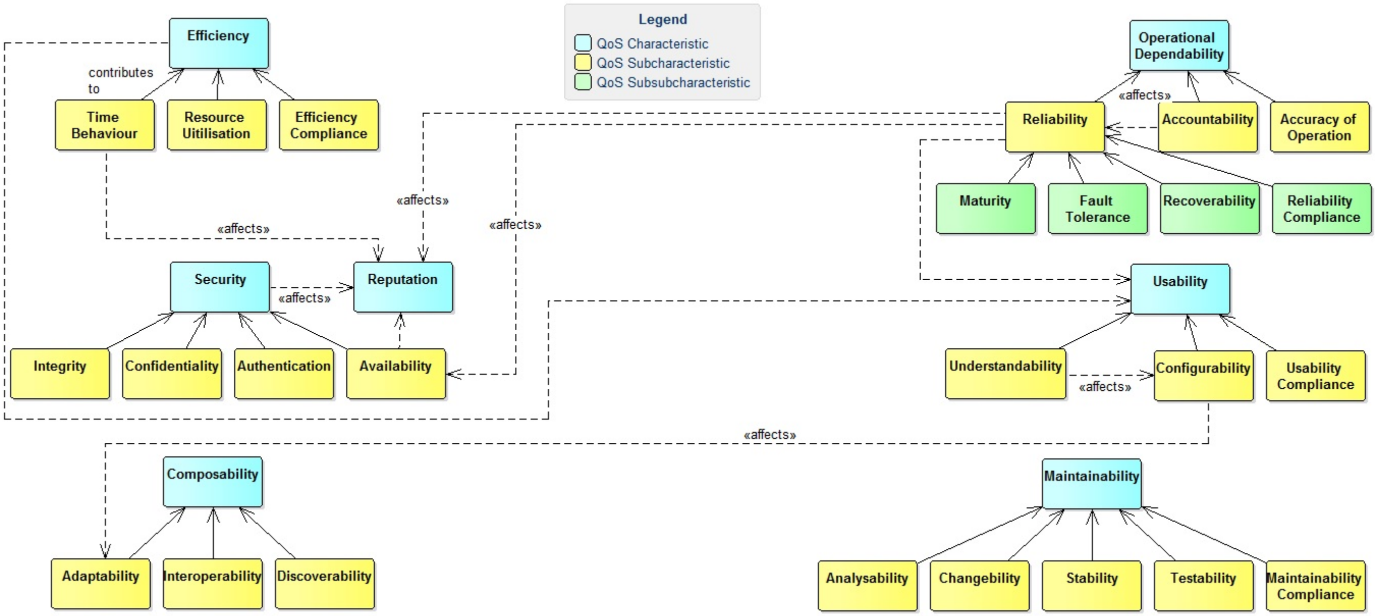

Figure 2: SOA Quality Model

Therefore, first of all the *Information Need* with objectives, potential risks and expected problems is to be defined and documented properly. In terms of the application scenario presented in section III the objective is to assess the performance of the business process for calculating the offer in order to identify and resolve problems in time.

B. *Core measurement process*

The *Information Product* is the result of the execution of the *Core Measurement Process* as depicted in Fig. 3 (cf. [8]). The Information Need provides the input for the subprocess *Plan the Measurement Process* (planning stage), the subprocess *Perform the Measurement Process* (execution stage) generates the output, i.e., the Information Product. The process goal is to satisfy the Information Need. All concepts presented below directly or indirectly contribute to the Information Product.

C. *Concepts of the planning stage*

In the focus of our research work are *SOA Services* whose QoS is to be investigated. For this purpose, *Quality Attributes* are measured. In this context, the *Measurable Concept* outlines in an abstract way, *how* the attributes values are determined to satisfy the required Information Needs. In doing so, it references one or more sub-characteristics of the SOA QM.

For the application scenario described in section III, the process performance is to be determined first and then evaluated. The corresponding Measurable Concept is the calculation of the processing time. To do this, instantiation of a process and termination of the process instance are to be determined. The process identification represents the Quality Attribute to be measured, and the sub-characteristic, referenced by the Measurable Concept, is the Time Behavior.

In order to implement the Measurable Concept and to perform measurements of attributes, first of all *Measures* are

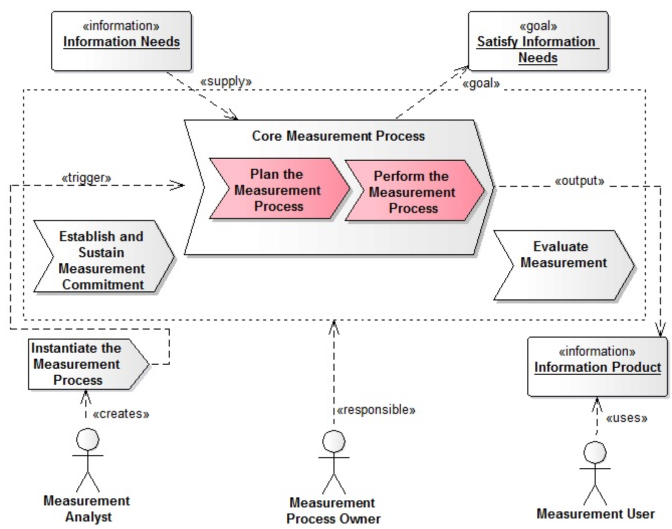

Figure 3: Process for determining the QoS (cf. [8])

to be specified. A Measure assigns each Quality Attribute a value on a *Scale* of a particular *Type*. The ISO/IEC-Standard 15939 provides 3 different types of Measures, namely Base Measures, Derived Measures, and Indicators respectively.

A *Base Measure* specifies by its *Measurement Method* how the value of a Quality Attribute is to be determined. It's always atomic and therefore independent of other Measures.

A *Derived Measure* uses one or more Basic Measures or other Derived Measures, whilst the *Measurement Function* specifies the calculation method and thus the combination of the Measures used.

For the application scenario illustrated in section III, the Basic Measures process instantiation t_{inst} and process in-

stance termination t_{term} are specified. The identification of the processes instance $piID$ represents the Quality Attribute measured by t_{inst} and t_{term}. As the Measurement Method, we select the time of the start and end event respectively. The Derived Measure processing time of the instance T_{Proc} will be calculated by the Measurement Function
$T_{Proc}(piID) = \Delta t = t_{term}(piID) - t_{inst}(piID)$.

Finally, an *Indicator* is a qualitative evaluation of Quality Attributes, which directly addresses the issue raised in the Information Needs. Indicators always use a nominal scale with qualifying values and thus show if necessary action or the need for further root cause analysis. An Indicator is derived from other Quality Measures, i.e., Base and Derived Measures, and Indicators. The combination of the Quality Measures used and the method of calculation is based on an *Analysis Model* in conjunction with *Decision Criteria* using thresholds and target values.

For the application scenario illustrated in section III, the indicator adequacy of the processing time of a process instance $SLoT_{Proc}(T_{Proc})$ according to table I:

TABLE I. ADEQUACY OF THE PROCESSING TIME

T_{Proc}	$SLoT_{Proc}$
$\in (0, 3000ms]$	high
$\in (3000ms, 7000ms]$	medium
$\in (7000ms, \infty)$	low

D. Concepts of the execution stage

After the concepts of the planning stage have been presented, now those of the execution phase will be explained briefly (subprocess *Perform the Measurement Process*, depicted in Fig. 3). Section V will discuss their conceptual implementation more detailed.

The actual measuring procedure, i.e., the execution of the instructions for determining the value of a Quality Attribute, is called *Measurement*. Hereby, *Measurement Results* are created, collected in a container, namely *Data*, which is inserted into a *Data Store*.

The measurement system comprises different supporting software components, which are conceptually presented in section V. The *QoS Measurement* performs the instructions specified in the Measurement Method or Measurement Function respectively, to generate the Measurement Results for further processing. The *QoS Analyser* performs the statistical analysis and evaluation of the collected data and creates the Information Product. The *QoS Reporting* makes the Information Product available to the *Measurement User* (cf. Fig. 3)

E. QoS Measurement Information Model

We designed a domain-specific language to specify the values of the concepts introduced above according to the Information Need. This specification document is referred to as *QoS Information Model (QoS IM)*. The aim of this approach is to automate the measurement process by the generation of artifacts required by the QoS system to execute a measurement.

The QoS IM consists of an abstract and a concrete section. In the abstract section, the concepts of the Planning Stage and partly the Execution Stage are specified. In the concrete section, the implementation specific definitions are done. Since our QoS-System is based on a complex event processing (CEP) approach, the specification of events, agents and rules is subject of this section.

A sophisticated XML Schema was developed to realize the domain-specific language. We opted for XML as a universally accepted standard that is highly flexible, platform and vendor independent and supported by a wide variety of tools. In a follow-up project an XText-based tool will be developed that generates the (XML) QoS IM from a (XText) source code.

Its semantic model is shown in Fig. 4. The following rules for modeling apply:

- Concepts are mapped to XML elements (graphically represented by UML classes).
- Details of a concept are mapped to XML attributes of the owning element (graphically represented by UML instance variables).
- If possible, relationships between concepts are mapped to element hierarchies (graphically represented by UML associations).
- Otherwise they are mapped to constraints (i.e. keyrefs) (graphically represented by UML dependencies).

V. MEASUREMENT CONCEPT

In section IV, a QoS IM based upon a SOA QM is described. To execute a specific QoS IM (and thus subprocess ”Perform the Measurement Process”) an execution platform is needed. This platform and the underlying QoS architecture is given in this section. First reasons for choosing this specific architecture are discussed shortly. Furthermore an overview is shown, detailing in the central agent concept and CEP.

A. Design Decisions

As described above the goal of the measurement concept is to provide the execution platform for a specific QoS IM. Therefore basic design criteria for the measurement concept are derived from the QoS IM. Furthermore quality requirements are given, which also have to be considered in the architecture design. These criteria are:

- measurement of Quality Attributes as described by QoS IM,
- flexibility of measurement and computation,
- low impact (modification, performance, etc.) onto SOA components.

The proposed Measurement Concept is based upon a general architecture given in [3]. The basic idea is to separate the measurement (e.g. sensors, agents, etc.) and ”analysis and statistics” functionality into different modules. This separation opens the opportunity to cater each module to their specific functional and quality requirements.

 ISBN: 978-1-61208-528-9

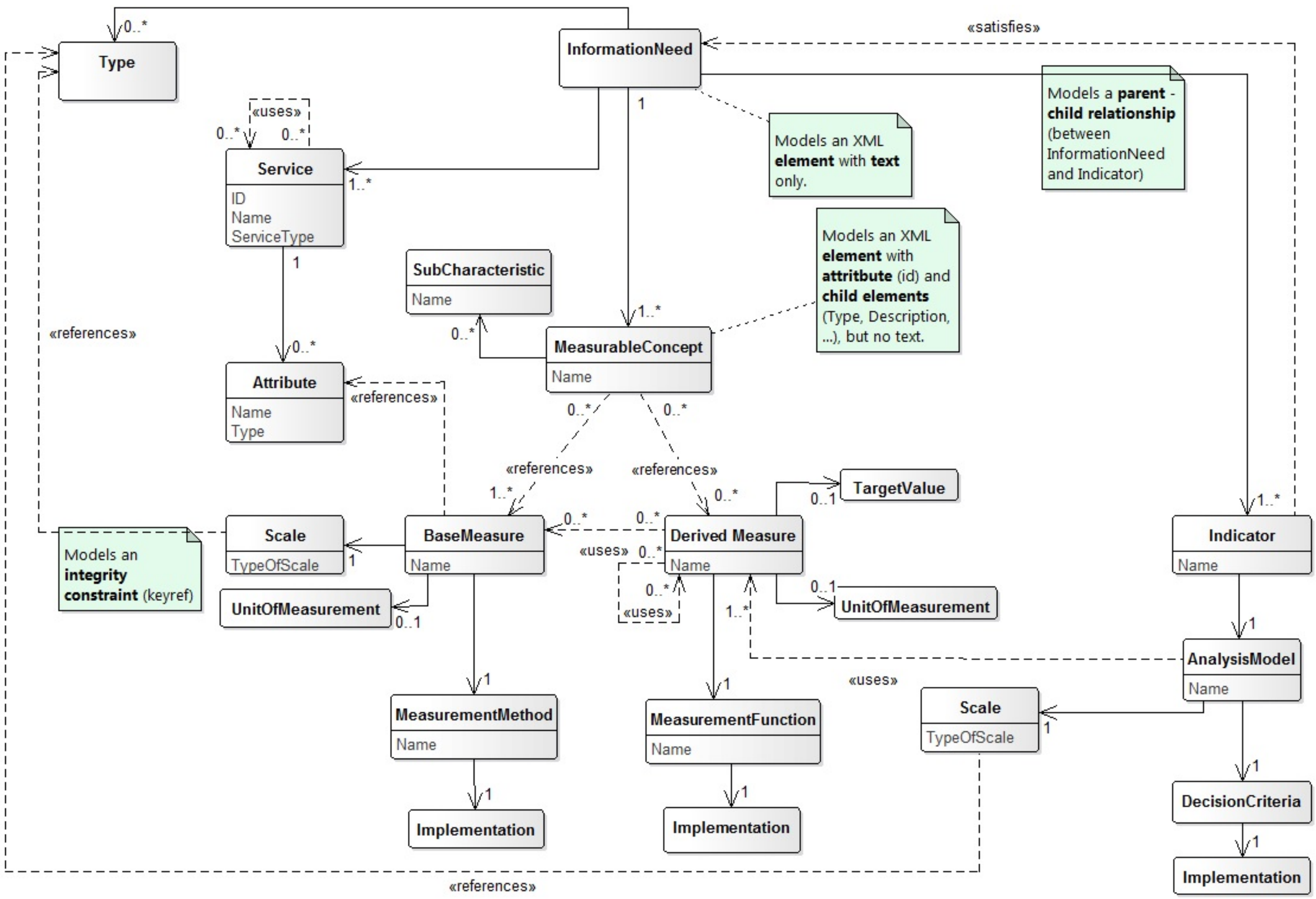

Figure 4: QoS Measurement Information Model (QoS MIM)

Overall the given general architecture already fulfills the requirement to measure Quality Attributes and provide the needed evaluations to produce Measurement Results and Information Products.

The measurement module has to provide the QoS System with information about the observed service. To provide the needed flexibility a sensor has to be placed into it. To keep the impact onto the SOA at a low level an agent based approach was chosen. Agents capsule the needed parsing and computation and thus can be easily integrated into arbitrary SOA modules. Furthermore minimizing the performance impact (through threading, non-blocking, etc.) can be integrated into the agents.

The ”analysis and statistics” module does not have these strict requirements on performance impact. Flexibility of computation and measurement execution is the main quality requirement. Thus a platform approach was chosen. Basically artifacts generated through the QoS IM are placed into the QoS platform and executed.

B. Overall system architecture

On a high level the QoS system splits the measurement agents and further processing (QoS platform) into different components. This approach allows to easily split these components into different processes to comply to the quality requirements. While the measurement agents (encapsulating the agent concept) represents the client component, the server component is represented by the QoS platform and contains the CEP engine and further analysis processing. Fig. 5 shows a high level overview of important components and their relationships.

The general purpose of the measurement agents is to emit specific events based on the defined Base Measures. As described in section IV events are emitted, e.g., for process instance instantiation/termination. In general, concepts for agent implementation can be categorized by agent location and time of execution (cf. [5] and [4]). To measure a specific process instance agents can be placed into corresponding SOA service calls thus measurement agents are only logically placed into the QoS System component. One and currently used approach is to use the concept of interceptors, which offers a low modification impact and can deliver precise Measurement Results.

The QoS Platform consist of several components, most notable the QoS Measurement and QoS Analyzer. In general the purpose of these modules are to collect, clean and compute

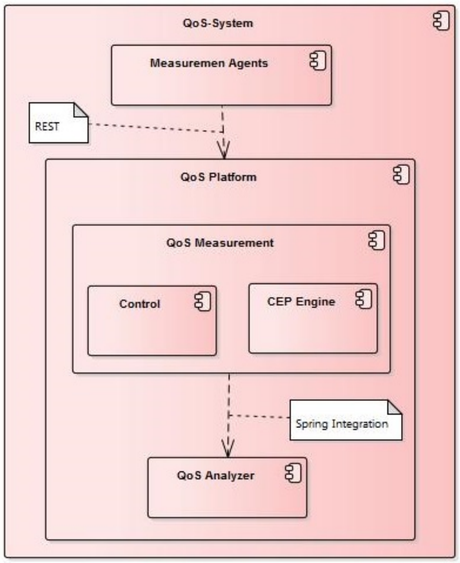

Figure 5: The QoS architecture

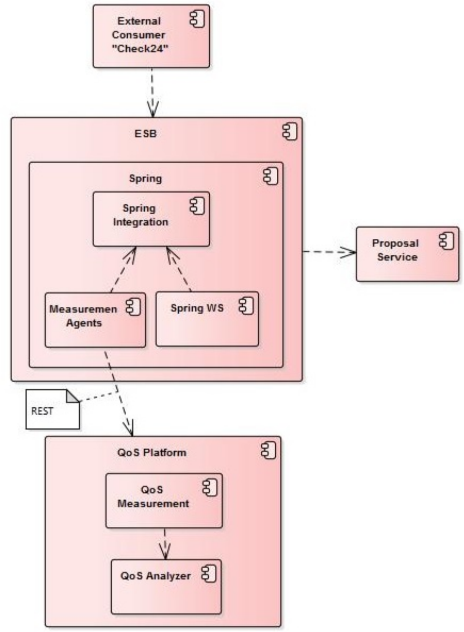

Figure 6: Applied QoS architecture

the emitted events and provide further analysis of stored Measurement Results (specifically stored as complex events).

Before any event is given to the Measurement Method, it will be handled by the control module. Purpose of this module is event routing, general cleaning steps and an optional filter step. Cleaning (or formatting) events in the analyzer is needed because measurement agents are placed in the monitored system, thus shall minimize their performance impact. The Measurement Method is implemented as a CEP rule executed by the engine and emits complex events for further near real-time processing and long term analysis.

The QoS Analyzer module provides a basis for statistical analysis and evaluations. Every complex event is stored into a Data Store implemented as a relational database. The different analysis and evaluations defined by Derived Measures and Indicators are implemented through SQL and plain Java. Furthermore the module provides an interface to the computed Information Product.

C. Applying the described measurement concept

In Fig. 6 the given measurement concept (QoS platform) is applied onto the application scenario thus providing the missing link between the QoS IM and the insurance based application scenario. In this example a simplified scenario is used consisting only of an external ”Check24” mock-up service (representing a simple consumer), the central ESB and the proposal service (which represents the producer). The task of the measurement model and thus the concept is to measure the processing time of this service and to compute the Information Product for further evaluations.

To measure this service the measurement agents, defined through base measures, are placed directly into the ESB. This offers a measurement independent of service location and different load balancing scenarios. To minimize the integration effort functionality given by Spring Integration is extensively used (especially the interceptors for message queues). In this simplified example base measures (and thus the agents) only determine service call start / end times and announces these to the QoS platform. Furthermore the agents try to minimize their performance impact by using non-blocking techniques and performing only necessary parsing steps (e.g. service call id’s, etc.). These will be shown in detail in further publications.

As described above the QoS platform performs further cleaning and processing steps to compute the QoS IM indicators ($SLoT_{Proc}(T_{Proc})$) and provides these to downstream systems (e.g. reporting, presentation, load balancing, etc.).

VI. CONCLUSION AND FUTURE WORK

The presented approach for monitoring a distributed SOA environment is a promising path to take: The SOA QM is aiming to follow the ISO/IEC-Standard 15939 (cf. [8]), which enables a wide range of use cases. The Measurement Concept outlines an execution platform for the specific QoS IM, which should cause minimal impact on the SOA environment. The separation of Measurement Agents and QoS-Analyzer allows lightweight agents on the one hand and a very capable analyzer component on the other hand.

The still ongoing work of applying the QoS System to an application scenario relevant to our partner in the insurance industry (the ”Check 24 process”), will provide evidence of the practical usability of the created framework. In this paper the framework and the corresponding plattform are applied onto a basic, business relevant scenario (the proposal service). Furthermore it is planned to apply these technique to the more complex process ”Angebot erstellen” (”create individual proposal”) of the VAA thus implementing a more complex scenario. It is expected that the monitoring system will help to discover potential bottlenecks in the current system design of our partners distributed services and therefore creating high value in the process of solving these issues.

In future work, the actual measurement and analysis of the results are to be done. It is also planned to apply these results onto cloud based environments.

REFERENCES

[1] J. Brutlag, "Speed Matters for Google Web Search," Google Inc., Mountain View, 2009, [Online]. URL: http://services.google.com/fh/files/blogs/google_delayexp.pdf [accessed: 2016-12-26].

[2] Dynatrace LLC, "Dynatrace Application Monitoring," [Online]. URL: https://www.dynatrace.com/de/products/application-monitoring.html [accessed: 2016-12-26].

[3] A. Wahl, A. Al-Moayed, and B. Hollunder, "An Architecture to Measure QoS Compliance in SOA Infrastructures," Service Computation, 2010, pp. 27-33.

[4] E. Oberortner, U. Zdun, and S. Dustdar, "Patterns for Measuring Performance-related QoS Properties in Service-oriented Systems," Pattern Languages of Programs Conference, 2010, pp. 1-21.

[5] E. Oberortner, S. Sobernig, U. Zdun, and S. Dustdar, "Monitoring Performance-Related QoS Properties in Service-Oriented Systems: A Pattern-Based Architectural Decision Model," EuroPLoP, 2011, pp. 1-37.

[6] M. Azuma, "SQuaRE: the next generation of the ISO/IEC 9126 and 14598 international standards series on software product quality, " in: Proc. of European Software Control and Metrics (ESCOM), 2001, pp. 337-346.

[7] H. Balzert, "Lehrbuch der Softwaretechnik: Softwaremanagement", Springer Spektrum, Heidelberg, 2008.

[8] ISO - International Organization for Standardization (ed.), "ISO/IEC 15939:2007 - Systems and software engineering - Measurement process," 2007.

[9] ISO - International Organization for Standardization (ed.), "ISO/IEC 25010:2011 - Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models", 2011.

[10] F. Garcia, et al., "Towards a consistent terminology for software measurement," in: Information and Software Technology, vol. 48, 2006, No. 8, pp. 631-644.

[11] DIN NIA, "ISO/IEC 25000 System und Software-Engineering - Qualitätskriterien und Bewertung von System- und Softwareprodukten (SQuaRE - Leitfaden für SQuaRE," [Online]. URL: http://www.din.de/de/mitwirken/normenausschuesse/nia/normen/wdc-beuth:din21:204260933 [accessed: 2017-01-01].

[12] B. Schroeder, "On-Line Monitoring: A Tutorial," IEEE Computer, 28(6), pp. 72-80, 1995.

[13] S. Schwiderski, "Monitoring the Behavior of Distributed Systems," PhD thesis, Selwyn College, University of Cambridge, University of Cambridge, Computer Lab, Cambridge, United Kingdom, 1996.

[14] Nagios.ORG, "Nagios Core Editions," [Online]. URL: https://www.nagios.org/ [accessed: 2016-12-26].

[15] T. Bergemann, A. Hausotter, and A. Koschel, "Keeping Workflow-Enabled Enterprises Flexible: WfMS Abstraction and Advanced Task Management," in: 4th Int. Conference on Grid and Pervasive Computing Conference (GPC), 2009, pp. 19-26.

[16] C. Gäth, et al., "Always Stay Agile! – Towards Service-oriented Integration of Business Process and Business Rules Management," in: The Sixth International Conferences on Advanced Service Computing (Service Computation), IARIA, Venice, Italy, 2014, pp. 40-43.

[17] A. Hausotter, C. Kleiner, A. Koschel, D. Zhang, and H. Gehrken, "Always Stay Flexible! WfMS-independent Business Process Controlling in SOA," in: IEEE EDOCW 2011: Workshops Proc. of the 15th IEEE Intl. Enterprise Distributed Object Computing Conference, IEEE: Helsinki, Finnland, 2011, pp. 184-193.

[18] N. W. Paton (ed.), "Active Rules for Databases," Springer, New York, 1999.

[19] ACT-NET Consortium, "The Active DBMS Manifesto," ACM SIGMOD Record, 25(3), 1996.

[20] A. Koschel and R. Kramer, "Configurable Event Triggered Services for CORBA-based Systems," Proc. 2nd Intl. Enterprise Distributed Object Computing Workshop (EDOC'98), San Diego, U.S.A, 1998, pp. 1-13.

[21] M. Schaaf, I. Astrova, A. Koschel, and S. Gatziu, "The OM4SPACE Activity Service - A semantically well-defined cloud-based event notification middleware," in: IARIA Intl. Journal On Advances in Software, 7(3,4), 2014, pp. 697-709.

[22] T. Frotscher, G. Starke (ed.), and S. Tilkov (ed.), "Der Webservices-Architekturstack," in: SOA-Expertenwissen, Heidelberg, dpunkt.verlag, 2007, pp. 489-506.

[23] F. Curbera, R. Khalaf, and N. Mukhi, "Quality of Service in SOA Environments. An Overview and Research Agenda," in: it - Information Technology 50, 2008(2), 2008, pp. 99-107.

[24] S.W. Choi, J.S. Her, and S.D. Kim, "QoS Metrics for Evaluating Services from the Perspective of Service Providers," in: Proc. of the IEEE International Conference on e-Business Engineering, Washington DC, USA : IEEE Computer Society (ICEBE'07), 2007, pp. 622-625.

[25] Z. Balfagih and M.F. Hassan, "Quality Model for Web Services from Multi-stakeholders' Perspective," in: Proceedings of the 2009 International Conference on Information Management and Engineering, Washington DC, USA : IEEE Computer Society (ICIME'09), 2009, pp. 287-291.

[26] M. Garcia-Valls, P. Basanta-Val, M. Marcos, and E. Estévez, "A bi-dimensional QoS model for SOA and real-time middleware," in: Intl. Journal of Computer Systems Science and Engineering, CLR Publishing, 2013, pp. 315-326.

[27] V. Krishnamurthy and C. Babu, "Pattern Based Adaptation for Service Oriented Applications," in: ACM SIGSOFT Softw. Eng. Notes 37, 2012(1), 2012, pp. 1-6.

[28] G. Wang, A. Chen, C. Wang, C. Fung, and S. Uczekaj, "Integrated Quality of Service (QoS) Management in Service-Oriented Enterprise Architectures," in: Proceedings of the 8th IEEE Intl. Enterprise Distributed Object Computing Conference (EDOC'04), Washington DC, USA, IEEE, 2004, pp. 21-32.

[29] B. Wetzstein, et al., "Monitoring and Analyzing Influential Factors of Business Process Performance," in: Proc. IEEE International Enterprise Distributed Object Computing Conference (EDOC'09), 2009, pp. 141-150.

[30] F. Rosenberg, C. Platzer, and S. Dustdar, "Bootstrapping Performance and Dependability Attributes of Web Services," in: Proc. International Conference on Web Services (ICWS'06), 2006, pp. 205-212.

[31] M. Schmid, J. Schaefer, and R. Kroeger, "Ein MDSD-Ansatz zum QoS-Monitoring von Diensten in Serviceorientierten Architekturen," in: PIK Praxis der Informationsverarbeitung und Kommunikation, 31 (2008) 4, 2008, pp. 232-238.

[32] GDV (Gesamtverband der Deutschen Versicherungswirtschaft e.V. – General Association of the German Insurance Industry), "Die Anwendungsarchitektur der Versicherungswirtschaft: Das Objektorientierte Fachliche Referenzmodell (The application architecture of the German insurance business – The functional object-oriented reference model," VAA Final Edition. Version 2.0., 2001, [Online]. URL: http://www.gdv-online.de/vaa/vaafe_html/dokument/ofrm.pdf [accessed: 2017-01-11].

[33] M. Varela, L. Skorin-Kapov, F. Guyard, and M. Fiedler, "Meta-Modeling QoE", PIK-Praxis der Informationsverarbeitung und Kommunikation, 2014, Vol. 37(4), pp. 265-274.

[34] R.W. Maule and W.C. Lewis, "Performance and QoS in Service-Based Systems", Proc. of the 2011 IEEE World Congress on Services, IEEE Computer Society, 2011, pp. 556-563.

[35] S.M.S. da Cruz, R.M. Costa, M. Manhaes, and J. Zavaleta, "Monitoring SOA-based Applications with Business Provenance", Proc. of the 28th Annual ACM Symposium on Applied Computing (ACM SAC), ACM, 2013, pp. 1927-1932.

Underground Facility Monitoring Services for Detecting Road Subsidence

Kwangsoo Kim, Dong-Hwan Park, Jaeheum Lee, and Inwhan Lee
UGS Convergence Research Division
Electronics and Telecommunications Research Institute
Daejeon, Republic of Korea 34129
Email: {enoch, dhpark, ljh, ihlee}@etri.re.kr

***Abstract*—This paper presents underground facility monitoring services to detect road subsidence. Each service includes a wireless sensor network, middleware, an analyzer, and a visualizer. The network, equipped with many sensor nodes and an underground safety-access point (UGS-AP), acquires sensing values on the physical condition of underground facilities and transmits them. The middleware manages, stores, and provides the values to the analyzer and the visualizer. The analyzer evaluates the status of the facilities by using the sensing values and generates a subsidence risk index (SRI) in a certain area, which indicates the risk level. The SRI is the result of the aggregation of the statuses generated from all underground facilities included in the area. Each SRI is classified in a category (low, medium, and high risk level). The visualizer displays the SRI with geometric objects and imagery data. This service will enhance the ability to increase public safety by monitoring the underground space and help explain the causes of the road subsidence by analyzing the sensing values.**

Keywords–Underground safety; Subsidence risk index; Underground facility.

I. INTRODUCTION

The industrial revolution brought about many changes in the social life of people. One of the most important things was an excessive population shift from rural to urban areas. Rapid population movement changed a rural society where most people lived on farms to an urban society where most people lived in cities. According to a report published by the Economist, about 64 % of the developing regions and 86 % of the developed regions will be urban areas by 2050 [1]. Consequently, 66 % of the world's population will dwell in urbanized areas by 2050. The rapid growth of urban population developed infrastructure to increase health and to support a comfortable life for the city dwellers. The infrastructure contained everything from water supply and sewer systems to road, power, and rail networks. Many different types of infrastructures were installed underground. Water pipes and sewer pipes are primarily underground and out of sight. There are several reasons that human beings use and develop underground space.

- The lack of surface place leads to use the underground space.
- Underground space is used not to spoil the aesthetic feature of city.
- Underground space is used to provide natural protection for whatever is placed in it.
- Underground space is used to protect the surface environment.

However, it is no doubt that the infrastructure of many big cities around the world is aging and failing. The current outdated and aging infrastructure is creating many problems. Each year, many water pipes broke and sewer pipes cracked in big cities. For example, water pipes of New York City were constructed more than 100 years ago [2]. The aged pipes resulted in frequent and disruptive breaks. Since 1998, more than 400 water pipe breaks all over the city have happened every year. Also, road subsidence caused by the breaks and cracks of underground infrastructures has emerged in downtown areas in Korean cities. This situation is considered as a very serious social problem because some people fell into the holes created by the subsidence. According to a report published by Seoul, road subsidence appeared in 3,328 locations from 2010 to 2014 [3]. Its occurrence trend is shown in Figure 1. We can clearly see that road subsidence had happened more than 400 times every year. Figure 2 shows the heatmap that highlights the occurrence of the road subsidence in a certain region. We can clearly see that road subsidence occurs across Seoul. The heatmap is based on dividing the entire area into 50m $\times$ 50m grids. The metric is the sum of the number of road subsidence occurrences in each grid. The red indicates that many road subsidence occurred in the grid. By several investigation reports on the road subsidence, the road subsidence in downtown areas is formed by empty space in the underground. Breaks in water supply pipes, cracks in sewer pipes, and poor backfilling are identified as the main causes of the empty space. In addition, the subsidence appeared in other cities such as Busan, Incheon, Daejeon, Gwangju, Jeonju, Suwon, Suncheon, Yongin, Andong, etc.

Unfortunately, the failures of the underground infrastructures are not predictable as well as the failure detection is very difficult. There are several reasons that make the failure detection difficult.

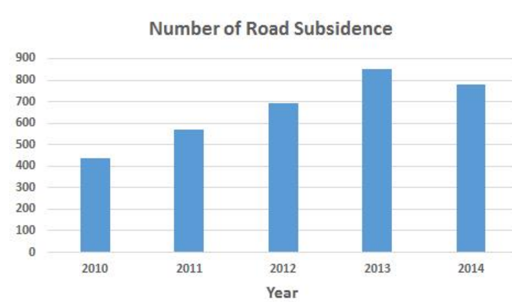

Figure 1. Number of road subsidence by year in Seoul (Source: Seoul City).

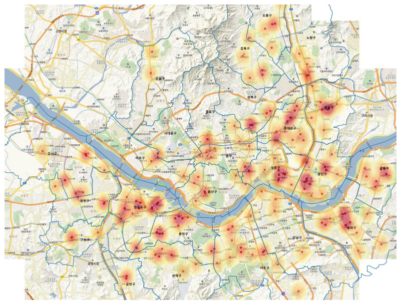

Figure 2. Road subsidence in Seoul shown as heatmap on Google map (Data source: Seoul City).

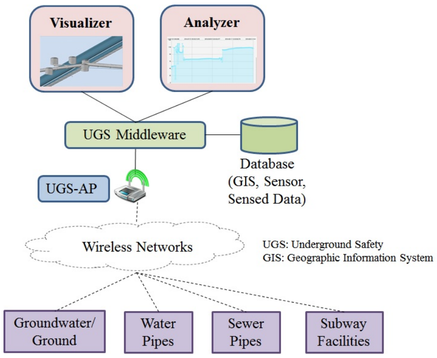

Figure 3. Architecture of underground facility monitoring service.

- The underground space and infrastructure in it are invisible.
- The failures suddenly happen without early warning signs.
- We do not even know where the infrastructure is below ground.
- It is not easy to explain what exactly leads to the failures.

Although the road subsidence raises the public safety concerns in Korea, investigators are struggling to find out the cause of the failures because there is no evidence to explain the process of the failures. We have developed an underground monitoring system to recognize when a risk suddenly happens in the underground area and to provide the evidence. To achieve this goal, this system collects and monitors the state changes of various kinds of underground facilities and utilities including water pipes, sewer pipes, metro lines, metro stations, ground, and groundwater levels [3], [4]. The monitoring system monitors the current state of the underground infrastructure and also records the history of its state change. The development of novel methods for non-destructive evaluation of underground infrastructure is one of the interesting research areas. Figure 3 shows the architecture of the underground facility monitoring service which consists of underground facilities, wireless networks, UGS-AP (Access Point), UGS (Underground Safety) middleware, database, and visualizer [3]. To acquire sensing data related to the state changes, various types of sensing devices are attached to those underground facilities and their sensing values are sent to the visualizer through the UGS-AP and the UGS middleware. The UGS-AP provides various communication methods including Ethernet, long term evolution (LTE), and WiFi. The UGS middleware is responsible for collecting, storing, and managing sensing data. The visualizer displays the sensing values, sensor nodes, and appearances of all underground facilities. This module uses GIS technology to similarly represent the appearance of the individual facility based on the actual shape.

The rest of this paper is structured as follows. Section II introduces wireless sensor networks that capture the states of each facility and transmit them. Section III introduces the UGS middleware that collects, stores, and manages sensed data and GIS data. Section IV introduces the analyzer which evaluates the status of each underground facility. Section V introduces the visualizer which displays risk indexes, sensing data, and geographic data. Finally, Section VI concludes this paper.

II. Wireless Sensor Networks

A wireless sensor network (WSN) is used to collect the state changes of underground infrastructures. The network includes two kinds of devices: an UGS-AP and sensor nodes.

These devices communicate with each other wirelessly and form a star topology to transfer data fast without collisions. This data transmission scheme gives the network better performance. An UGS-AP interconnecting a WSN and an IP-based network includes various communication schemes such as Ethernet, WiFi, and long term evolution (LTE). A sensor node involves one or more sensors which collect the states of underground utilities. Sensor nodes are attached to water pipes, sewer pipes, and subway lines and installed in tube wells. Each sensor node transmits its sensing values to the UGS-AP which it connects to.

Underground facilities produce various types of sensing values according to their properties. A sewer pipe produces still images and videos which show its internal state like a crack. Those values are captured by a sewer inspection camera attached to a sewer robot. A water pipe produces leak noise and motion changes such as pitch (lateral axis) and roll (longitudinal). The noise and motion are captured by a acoustic sensor and a gyroscope, respectively. A subway line and station produce videos, still images, stress, acceleration, and the amount of influent water. Videos and still images are captured by a tunnel inspection camera. The acceleration and stress are captured by a strain gage and an optical fiber-based sensor. The amount of water is captured by a flowmeter. A tube well produces water level, water temperature, water conductivity, water turbidity, soil temperature, soil conductivity, and soil moisture. The water level is captured by a water level sensor. The temperatures are captured by temperature sensors. The conductivities are captured by conductivity sensors. The moisture is captured by a moisture sensor. All sensors are installed in one sensor node. Most of the sensing values are sent to the database through the UGS middleware and the WSN; however, still images and videos are uploaded into the storage through the Internet.

III. UGS Middleware

UGS Middleware (UGS-M) is located between some applications including the visualizer and the UGS-AP which receives sensing values from a large number of sensor nodes. UGS-M plays a role in collecting data from the UGS-AP, validating them, transmitting them to the applications, storing them to the database, providing access control to sensing devices, and providing an abstraction on various specifications of communication protocols and sensing devices. The abstraction enables an application to reduce the heterogeneity of different devices and to use them effortlessly. UGS-M provides two data formats, binary and XML, to exchange data with the UGS-AP. To achieve this goal, we classify the required functions of UGS-M in six functional components. They are a communication manager, resource manager, monitoring manager, data translator, sensing data manager, and a service interface [3]. UGS-M uses a RESTful interface as a service interface for applications.

We use Kairos [5] as a data storage and manager. Kairos is a spatial database management system (DBMS) which manages geometry data types as well as typical data types such as various numeric and character types at the same time. As Kairos is a main memory based DBMS, it shows high performance on spatial and typical query processing. It provides the typical database functions such as storage management, indexing, query processor, user management, etc. It also provides GIS engine including spatial data types, spatial operations,spatial indexing, spatial analysis, etc. Kairos stores geometric objects such as water pipes and sewer pipes, their attributes such as diameters and materials, and sensing values.

IV. Analyzer

An analyzer evaluates the status of the facilities by using the sensing values and generates a SRI in a certain area, which indicates the risk level. The SRI is the result of the aggregation of the statuses generated from all underground facilities included in the area. The SRI is represented as a numerical value as well as graded in a category (low, medium, high risk level). A lot of properties of each underground utility are used to estimate its status value. We use four status values generated from water pipes, sewer pipes, subway lines, and ground, one value from one utility. The evaluation of the status values related to underground utilities must be performed with different methods, according to their properties.

V. Visualizer

A visualizer is implemented as a client-server application for a Web-based service. In this model, graphical user interface (GUI), forms, and business logics are included in a client application and GIS engine for 2D/3D objects, data query and processing, and user management are included in a server application. Figure 4 shows the data flow in the visualization service. The Web server is accessed with a Web browser on a client and can be used by a lot of clients. The client application is running on both a personal computer and a mobile device based on Android. Each client requests 3D geometry objects, their attributes, and sensing data to the Web server. The Web server separates the requests into two groups. One is for a 3D geometry and its attribute; the other is for sensing data, 2D geometry, and its attribute. The former is processed in the Web server; the later is sent to UGS-M.

A. Architecture of Visualizer

The visualizer is designed and implemented to provide a Web-based service using Java technologies. Its architecture is shown in Figure 5. This visualizer is implemented based on e-government framework. The visualizer consists of the presentation, business, data, and base layer which has its own functionality. Each layer is implemented by Java technologies to ensure portability and flexibility of the service.

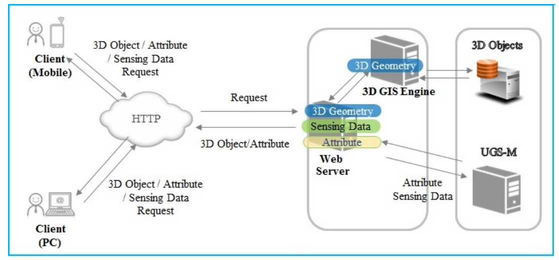

Figure 4. Data flow among components in visualizer.

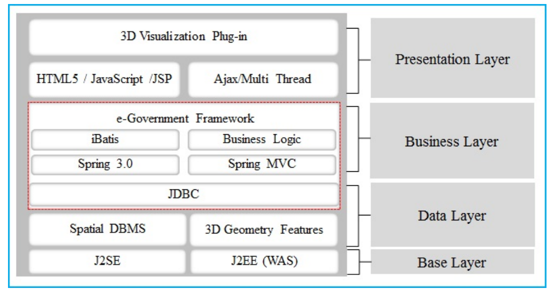

Figure 5. Architecture of visualizer.

1) e-Government Framework: The e-Government framework provides a standard operating environment for developing Java-based information systems. The Korean government has developed this framework to increase the quality of governmental services. As it also is open to the public sector, the framework is widely used to increase the efficiency of the development of information systems in the sector and to shorten the development period. The framework has several advantages such as reuse common features, increase interoperability, standardization, and openness. Its core technology is the integration of Spring and iBatis. Spring is a framework of various frameworks; iBatis provides automatic connectivity between databases and Java objects. We used this framework in the business layer and data layer to increase the efficiency of the data connection and the data processing flow.

2) Presentation Layer: The presentation layer displays 2D/3D geometry objects, their attributes, SRI values, and sensing data. To display this data, this layer provides the GUI of this application and involves a series of forms for client interaction.

3) Business Layer: The business layer implements the business logics and policies of this application. This layer represents the business rules that control the data flow concerned with the retrieval, processing, transformation, and management of application data. This layer also ensures data consistency and validity.

4) Data Layer: The data layer provides access to 3D feature files and spatial databases. This layer consists of the definitions of database schemata, tables, and columns. This layer also includes the program logic which is needed to navigate the database. This layer enforces rules regarding the database and access of data. The Java Database Connectivity

 ISBN: 978-1-61208-528-9

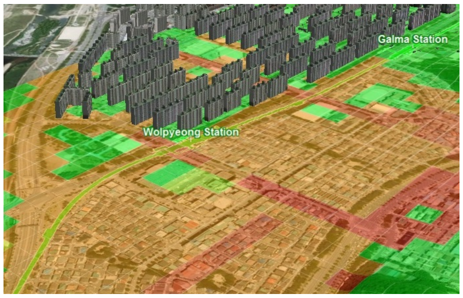

Figure 6. SRI grids.

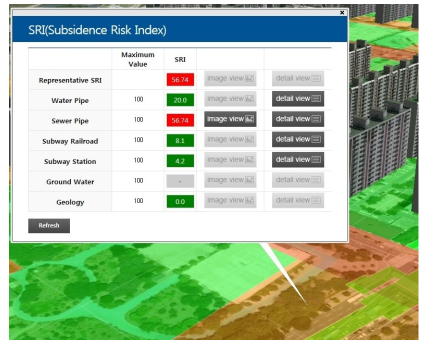

Figure 7. A grid cell and its status values.

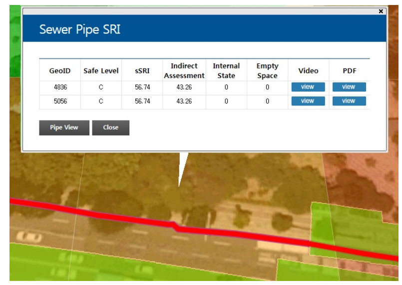

Figure 8. Representation of sewer pipes.

(JDBC) is used as a data access object.

5) Base Layer: The base layer provides the runtime environment to create this Web application. The Apache Tomcat is used as a Web application server which exposes business logic to client applications through HTTP.

B. Implementation Result

One of the most important objectives of the visualizer is to show the subsidence risk index (SRI) which represents the risk level of a certain region. The risk index is expressed with GIS data and imagery data provided by Daejeon Metropolitan City and V-World. Figure 6 shows the SRI grids in Seo-gu, Dajeon City where our testbed is installed. We divide the monitoring area into a lot of grid cells with 50m $\times$ 50m size. Each grid cell is colored in red, yellow, and green which indicate low, medium, and high risk, respectively. The SRI of a grid cell represents the highest risk value among the statuses of underground facilities in the grid. A risk values in Figure 6 is assigned to each grid cell by evaluating the state of each underground facility. Figure 7 shows the SRI value and the status values of a grid cell marked in red because the sewer pipe has the red state. These state values are displayed to check the cause of the risk level. The SRI value is rated from 0 to 1 and each status value is rated from 0 to $Maximum$. The SRI value is calculated as $SRI = MAX(S_w, S_s, S_m, S_g)$, where S_w, S_s, S_m, and S_g indicate the status of water pipes, sewer pipes, subway lines, and ground, respectively. The MAX function can be replaced by another such as MIN or SUM. Figure 8 shows the sewer pipes and their states in the grid cell. This space includes the dangerous sewer pipes that are depicted in red. The green object represents the subway station and line. The visualizer handles both two-dimensional geographic objects and three-dimensional objects.

VI. CONCLUSION

As many big cities have experienced serious social problems such as road subsidence and sinkholes, the evaluation and visualization of the underground facilities is very important to recognize the risk which suddenly occurs underground. This paper describes the characteristics of an underground facility monitoring service to detect the road subsidence. To detect the risk, this system attaches some sensor nodes to underground facilities such as sewer pipes, water pipes, and subway lines. It senses the states of facilities, collects the sensing values, generates risk values based on the statuses of facilities, and visualizes risk values and facilities represented by geographic objects. The risk values are displayed in a SRI grid cell to identify easily the risky areas.

We are planning to design and develop several analysis methods that evaluate the status of each facility and detect its state changes from the sensing values. We will also work to determine an optimal function and model to calculate the SRI value from the statuses of underground facilities. Furthermore, additional underground facilities and structures which affect the road subsidence will be considered to improve the reliability of the evaluation results. In addition, more research is needed because the exact expectation of the road subsidence is very difficult.

ACKNOWLEDGMENT

This work was supported by the National Research Council of Science & Technology (NST) grant by the Korea government (MSIP) (No. CRC-14-02-ETRI).

REFERENCES

[1] Economist, “Urban life: Open-air computers,” Special reports: Technology and geography, 2012.

[2] A. Forman, “Caution Ahead,” Report published by the Center for an Urban Future, Mar. 2014.

[3] K. Kim, D. H. Park, J. Lee, and S. I. Jin, “UGS Middleware for Monitoring State of Underground Utilities,” in Proceedings of the International Conference on Information and Communications Technology Convergence (ICTC) Oct. 28–30, 2015, Jeju, Rep. of Korea. The Korean Institute of Communcations and Information Sciences, Oct. 2015, pp. 1186–1188.

[4] K. Kim, D. H. Park, J. Lee, and I. Lee, “Underground Monitoring Systems using 3D GIS for Public Safety,” in Proceedings of the Eighth International Conference on Advanced Geographic Information Systems, Applications, and Services (GEOProcessing) Apr. 24–28, 2016, Venice, Italy. IARIA, Apr. 2016, pp. 24–28.

[5] H. Han and S. I. Jin, “A main memory based spatial DBMS: kairos,” in Proceedings of the 17th international conference on Database Systems for Advanced Applications - Volume Part II (DASFAA) Apr. 15–18, 2012, Busan, Rep. of Korea, Apr. 2016, pp. 15–18.

Quality of Service everywhere

Tadeus Uhl
Flensburg University of Applied Sciences
Flensburg, Germany
email: tadeus.uhl@hs-flensburg.de
Maritime University of Szczecin
Szczecin, Poland
email: t.uhl@am.szczecin.pl

Abstract— **Applications provided in the Internet of Things can generally be divided into three categories: audio, video and data. This has given rise to the popular term Triple Play Services. The most important audio applications are Voice over IP (VoIP) and audio streaming. The most notable video applications are Video Telephony over IP (VToIP), Television over IP (IPTV), and video streaming, and the service World Wide Web (WWW) is the most prominent example of data-type services. This paper elaborates on the most important techniques for measuring Quality of Service (QoS) and Quality of Experience (QoE) in Triple Play Services.**

Keywords-Triple Play Service; QoS; QoE; VoIP; IPTV.

I. INTRODUCTION

Quality of Service (QoS) is of crucial importance to modern digital networks, not least in the Internet, which is increasingly going under the name Internet of Things (IoT). The term QoS is becoming a household phrase, and has long been anchored in the definition of Next-Generation Networks in ITU-T Standard Y.2001 [1]. The International Telecommunication Union defines QoE as "The overall acceptability of an application or service, as perceived subjectively by the end-user" [2]. In November 2009, the European Parliament and the European Commission adopted the so-called Communications Packet, their directives 2009/136/EC [3] and 2009/140/EC [4] underlining the importance of QoS. Both QoS and QoE are to be monitored continually, and preferably automatically, in modern networks, and that means in the Internet of Things (IoT) too. EU research projects, such as Leone [5] and mPlane [6] and standardisation organisations, such as the Internet Engineering Task Force (IETF) [7], have been hard at work to achieve this. Monitoring the quality of service throughout the EU is, however, proving to be an especial challenge, given the dynamic structure of the Internet. New services are forever being introduced and older ones restructured. The IETF seeks to meet these challenges with the all-embracing Framework for Large-Scale Measurement of Broadband Performance (LMAP [7]) and, in keeping with the EU research project mPlane [6], promotes a flexible all-purpose solution.

Whilst these frameworks do contain fundamental concepts and specifications for the diverse areas of a distributed measuring system, such as the exchange and storage of acquired measurement data, by no means are they intended as complete systems. First of all, the suitable measuring equipment, storage units and results analysis software of various providers must be pieced together. In keeping with the trend of shifting more and more services into the cloud [8], the individual components of big-data systems have been designed to work flexibly in the cloud. Alongside these large-scale, complex yet adaptable systems there is a demand for specific, complete quality-assessment solutions that are easy to implement and use. Existing systems for monitoring Triple Play Services might benefit from the new possibilities of cloud computing. Since cloud-based solutions can usually be implemented without complicated precommissioning at the user's end, and the costs incurred through on-demand solutions are lower than long-term investments, the entry threshold will fall, and the use of indispensable measuring technology will become more appealing. For, despite the unequivocal recommendations, continuous measurement of quality of service is still not a universal practice, owing to the lack of suitable QoS/QoE measurement techniques for measuring environments.

Section II begins with an overview of QoS and QoE measurement techniques; then in Section III the QoS/QoE in the VoIP service will be described. Section IV provides information about QoS/QoE in the VToIP/IPTV service. Following that, Section V discusses QoS/QoE in the WWW service. The paper concludes with a summary and an outlook on future work.

II. OVERVIEW OF QOS AND QOE MEASUREMENT TECHNIQUES

In order to determine the QoS/QoE in a network, two models are generally used: a) dual-ended model and b) single-ended model; cf. Figure 1 [9]. In the case of the dual-ended model, two signals are used: a) the original signal and b) the degraded signal. These two signals are available uncompressed. For this reason, measurements can be carried out for both Quality of Experience (a subjective evaluation) and Quality of Service (an objective evaluation). In the case of the single-ended model, only the impaired signal (compressed) is available. This allows only an objective evaluation of QoS to be made. QoS measurement is referred to as "intrusive measurement" (offline) in the case of the dual-ended model, and as "non-intrusive measurement" (online) in the case of the single-ended model.

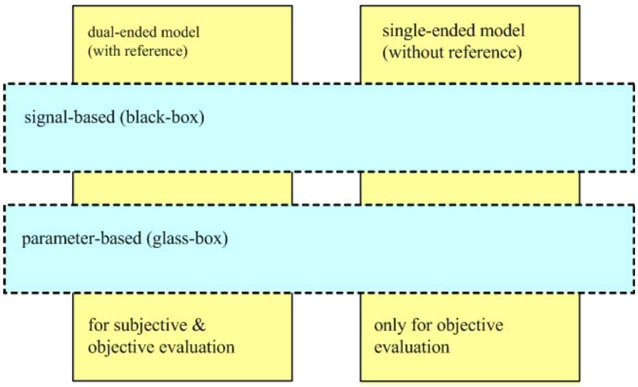

Figure 1. Overview of QoS and QoE measurement techniques.

Two measurement techniques can be used in the two models cited: a) signal-based and b) parameter-based measurement. The dual-ended model uses specialised algorithms to compare the input and output signals of signal-based measurements. In the case of the single-ended model, this comparison is made by using a reference signal. In both cases, the system to be assessed is treated as a "black box". When carrying out parameter-based measurements, a distinction is made between two types: a) "glass-box" and b) "black-box". In the first case, both the structure of the system to be assessed and the reaction of the individual system components to the reference signal are known. This knowledge is then taken into consideration in a suitable model. Additionally, the network parameters measured can be included in the calculation of the QoS. In the second case, not all details of the system to be assessed are known, so only the network parameters measured and the characteristic parameters for the respective service are taken into account.

The paper and the oral presentation at the Conference will review and discuss the application of common QoS/QoE measuring methods to Triple Play Services (audio/video/data).

III. QoS/QoE IN THE VoIP SERVICE

Figure 2 represents current QoS/QoE measurement techniques for the VoIP service. It is noticeable that several international standards touch on this area. The signal-based QoE measurement techniques PESQ [10] and POLQA [11] are very accurate; they are, however, time-consuming and can often only be implemented with a licence. That is why parameter-based QoS measuring methods are usually preferred in practice. The E Model [12], which was originally developed for circuit-switched telephone networks, has recently been adapted for use in IP networks to produce the modified E(IP) Model, released as a patent [13] in 2014.

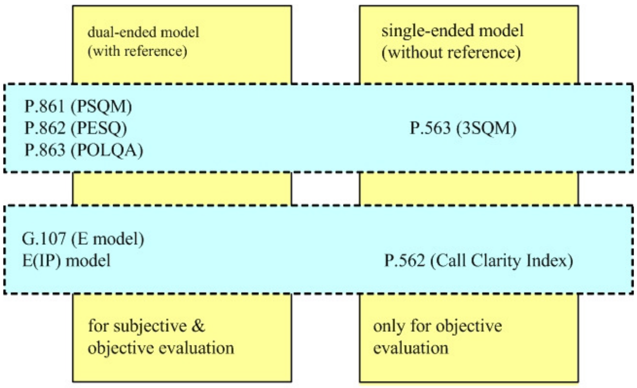

Figure 2. Overview of QoS and QoE measurement techniques for the VoIP service.

The modification in the E(IP) Model represents a restructuring of the two parameters *Ie* (Equipment Impairment Factor) and *Blp* (Packet-Loss Robustness Factor) to take account of impairment factors on the IP transport platform.

IV. QoS/QoE IN THE VToIP/IPTV SERVICE

Figure 3 shows established methods for measuring the QoS/QoE of the video component of the VToIP/IPTV service.

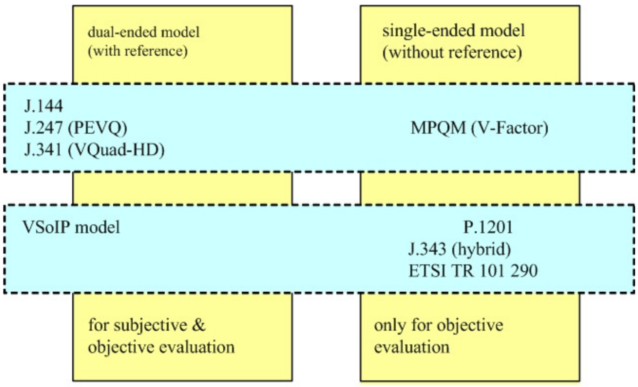

Figure 3. Overview of QoS and QoE measurement techniques for the VToIP/IPTV service.

Alongside several signal-based international standards there are also a few parametrised measurement techniques that work without a reference signal, not the least of these being the parametrised VSoIP Model [14], that is to be classified as a dual-ended model. Its make-up and mode of operation correspond to those of the E(IP) Model for the speech service. The parametrised QoS models are quick and easy to use, which is of great benefit in practice. In contrast, it takes minutes to measure the QoE of a HD video file using, say, the PEVQ [15] and the J.341 [16] method. Besides, they belong to the group of active (intrusive) methods of measuring QoE, which means that a connection

 ISBN: 978-1-61208-528-9

must first be established in an IP environment, and a reference signal must be sent and mirrored at the receiver's end. That is very time-consuming.

V. QOS/QOE IN THE WWW SERVICE

Figure 4 shows the most widely known techniques currently used to measure QoS/QoE in the WWW service.

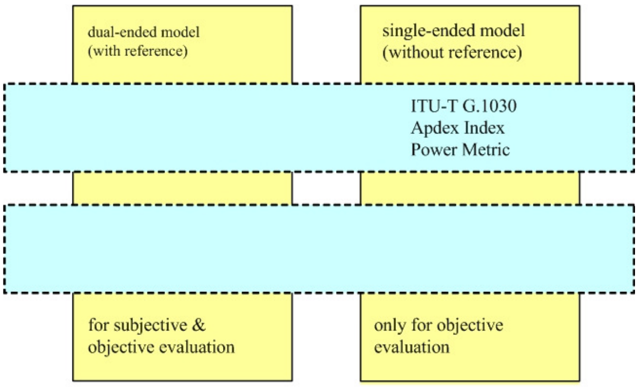

Figure 4. Overview of QoS and QoE measurement techniques for the WWW service.

It is immediately clear that there is only one standardised technique for measuring QoE in the WWW service: G.1030 [17]. There are two other QoS techniques in the single-ended model — Apdex Index [18] and Power Metric [19] — but they have not been standardised, meaning that there is an enormous need for further developments in this area, especially in view of the fact that the WWW service is one of the most widely used applications in the modern Internet of Things and accounts for the lion's share of traffic.

VI. CONCLUSION AND FUTURE WORK

This paper has reviewed and discussed briefly the application of common QoS/QoE measuring methods in Triple Play Services (audio/video/data). The strengths and weakness of the individual QoS/QoE measurement techniques have been spelt out. In practice it is highly beneficial to work with parametrised QoS models.

In the course of the related presentation in the track SERVQUAL [20], it will be demonstrated how the QoS measuring techniques that were described in the previous section are applied to the Triple Play Services. The results that were obtained from multiple series of QoS tests are presented in the form of graphs and then interpreted.

To summarise: a great deal of development work and practical implementation remain to be done in the field of QoS/QoE. New scientific concepts for QoS/QoE measurement techniques and systems are needed. Designers and engineers are faced with a mighty challenge!

REFERENCES

[1] ITU-T. *Definition of NGN*. [Online]. Avaiable from: http://www.itu.int/rec/T-REC-Y.2001/2017.02.01

[2] ITU-T. *Definition of Quality of Experience (QoE)*. [Online]. Avaiable from: http://www.itu.int/en/ITU-T/studygroups/2013-2016/12/Pages/default.aspx/2017.02.01

[3] EUR-Lex. *Directive 2009/136/EC of the European Parliament and of the Council of 25 November 2009*. [Online]. Avaiable from: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32009L0136/2017.02.01

[4] EUR-Lex. *Directive 2009/140/EC of the European Parliament and of the Council of 25 November 2009*. [Online]. http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32009L0140/2017.02.01

[5] EU Leon Project. *From global measurements to local management*. [Online]. Avaiable from: http://www.leone-project.eu/leone/2017.02.01

[6] mPlane Consortium. *mPlane – an Intelligent Measurement Plane for Future Network and Application Management*. [Online]. Avaiable from: http://www.ict-mplane.eu/2017.02.01

[7] IETF. *Large-Scale Measurement of Broadband Performance*. [Online]. Avaiable from: https://tools.ietf.org/wg/lmap/2017.02.01

[8] D. Felsmann, Cloud Computing – Basistechnologie, Architektur, Erfolgsfaktoren., Herausforderungen und die aktuelle Marktsituation. GRIN Editor, 2010.

[9] A. Raake, Speech Quality of VoIP. John Wiley & Sons, Chichester, 2006.

[10] ITU-T. *Standard ITU-T P.862: PESQ*. [Online]. Avaiable from:]http://www.itu.int/rec/T-REC-P.862/en/2017.02.01

[11] ITU-T. *Standard ITU-T P.863: POLQA*. [Online]. Avaiable from: http://www.itu.int/rec/T-REC-P.863/en/2017.02.01

[12] ITU-T. *Standard ITU-T G.107: E Model*. [Online]. Avaiable from:]http://www.itu.int/rec/T-REC-G.107/en/2017.02.01

[13] German Patent and Trade Mark Office. *E(IP) model,* German Patent no. 102010044727, September 2014.

[14] German Patent and Trade Mark Office. *VSoIP model,* German Patent Application. Current bibliography AKZ 10 2016 207 785.6. Mai 2016.

[15] ITU-T. *Standard ITU-T J.247: Objective perceptual multimedia video quality measurement in the presence of a full reference*. [Online]. Avaiable from: http://www.itu.int/rec/T-REC-J.247-200808-I/2017.02.01

[16] ITU-T. *Standard ITU-T J.341: Objective perceptual multimedia video quality measurement of HDTV for digital cable television in the presence of a full reference*. [Online]. Avaiable from: http://www.itu.int/rec/T-REC-J.341-201101-I/en/2017.02.01

[17] ITU-T. *Standard ITU-T G.1030: Estimating end-to-end performance in IP networks for data applications*. [Online]. Avaiable from: http://www.itu.int/rec/T-REC-G.1030/en/2017.02.01

[18] Apdex Alliance. *Application Performance Index - Apdex Technical Specification Version 1.1*. [Online]. Avaiable from: http://www.apdex.org/specs.html/2017.02.01

[19] T. Uhl, J. Klink, P. Bardowski, "New metric for World Wide Web Service Quality", Journal of Telecommunication and Information Technology, no. 2, pp. 50-58, Juno 2014.

[20] T. Uhl, "Quality of Servies everywhere", Presentaion submited on the Track SERVQUAL at the IARIA Conference 2017.

Selected Issues of Internet Access Service Quality Assessment

Janusz H. Klink
Telecommunications and Teleinformatics Department
Wroclaw University of Science and Technology
Wroclaw, Poland
e-mail: janusz.klink@pwr.edu.pl

***Abstract*— This paper presents selected issues related to Internet Access Service (IAS) quality assessment. Nowadays, it is not only the price, but quality, that influences the decisions of users regarding the choice of Internet Service Provider (ISP). According to European documents, users have a right to be informed of IT services offered by different providers. However, there is a problem: what, where and how to measure the service quality? The second issue is: how service quality is perceived by users and, finally, how to correlate these two different points of view. We discuss selected objective measures of the Internet Access Service and present measurement scenarios. The users' point of view and the subjective measure of quality is also presented. In the second part of the paper, we show how the users perceive Internet Access by the services they use. An example of building the Quality of Experience model for the WWW service is also presented.**

Keywords-Internet access; quality assessment; QoS; QoE; WWW quality model.

I. INTRODUCTION

In March 2010, the European Commission has launched a strategy entitled "Europe 2020", which sets the objectives for smart, sustainable and inclusive growth of the European Union by 2020 [1]. The Digital Agenda [2] forms one of the seven pillars of the strategy and defines the key enabling role that the use of Information and Communication Technologies (ICT) will have to play in Europe in future years. It is supposed to support a better quality of life, e.g., through better health care, safer and more efficient transport, a cleaner environment, new media opportunities and easier access to public services and cultural content. The Internet will be used as a vital medium for conducting business, as well as aiding work, play and communication between users. I will also be the center of the future economy, which will be based on network-based knowledge. It is assumed that by 2020 all Europeans will have access to Internet speeds of above 30 Mbps and at least 50% of the households will subscribe to Internet connections above 100 Mbps.

According to the European Commission, the digital sector grows seven times faster than other parts of industry. Thus, in September 2016, new Commission strategy documents on Connectivity for a European Gigabit Society were adopted [3]. They set a vision of Europe where "availability and take-up of very high capacity networks enable the widespread use of products, services and applications in the Single Digital Market". A vision of "Broadband Europe" assumes the building of the Gigabit Society by 2025 and relies on three main strategic objectives:

- Gigabit connectivity for all main of socio-economic drivers,
- uninterrupted 5G coverage for all urban areas and major terrestrial transport paths,
- access to connectivity offering at least 100 Mbps for all European households.

Consumer research has revealed that price is still the most important attribute taken into account when choosing an Internet access service for 20% of users [4]. The second decision-making factor is the data cap, i.e., the monthly limit on the amount of data a user can use with an Internet connection. Moreover, what happens when a user hits their limit is a very important issue. ISPs then engage in different actions such as slowing down data speeds, charging extra fees, or preventing further usage.

The next important factors, which may influence user attitude to an ISP offer, are service differentiation and traffic management such as prioritization, blocking or throttling. These practices aim to preserve the appropriate conditions for providing high-quality services. Nonetheless, in recent years these activities have raised questions about network neutrality, which assumes that all content and applications should receive equal treatment. Moreover, neutrality also means that providers neither impose nor discriminate in favor of using a particular type of technology [5][6].

Consumer awareness of network neutrality and traffic management is rather low. On one hand, most people have very little knowledge about these terms and, on the other hand, they do not see the influence of these issues on their Internet usage. As is shown in [7], consumers care very little for all the technicalities connected with data transport and the role of ISPs. Users are not interested in net neutrality or traffic management practices and instead are tied to their experience of traffic management effects.

Germany's "Wissenschaftliches Institut für Infrastruktur und Kommunikationsdienste" (WIK-Consult) study, which concentrates on contract-based consulting services for public and private institutions, asked a series of questions about the way consumers would respond to specific changes in the traffic management policies operated by their ISP, e.g., the

 ISBN: 978-1-61208-528-9

introduction of throttling on video traffic, or of data caps. A significant majority of respondents said that they would even change the provider in response to some significant changes in the traffic management policies of their ISP [4].

The issues mentioned above show a much higher interest of users in their ISP traffic engineering operations when these activities touch the concrete services and influence the users' experience. Nowadays, users not only trust the service level agreements of their providers, but also want to be able to check them.

The rest of this paper is organized as follows. In Section II, we present a general overview of IAS structure from the technical perspective as well as the users' point of view. Next, in Section III, the main parameters that may influence quality are discussed and the quality measurements of IAS, according to the present standards, are presented. Section IV describes the service quality issues as perceived by users. We underline the difference between objective quality measures and the subjective users' perception of different services used by them. We validate the need to build quality models for the most popular services and mention WWW browsing as one of them. In Section V, the Quality of Experience (QoE) model for the WWW service is discussed. We present the laboratory test-bed, measurement results and method of the model derivation. The paper ends with a conclusion and the plans for future work.

II. INTERNET ACCESS SERVICE

One of the major factors influencing the decision of users when choosing an ISP is the Internet Access connection throughput offered by the provider. However, there are many misunderstandings regarding this term. Physically, it is a combination of different connections and services that are needed to establish a functioning Internet access. Each of them can be treated as a separate service. Most users, however, treat Internet access as an access to the end-to-end services available on the Internet. A purely physical access to the Internet has no practical meaning to them. Thus, Internet access is generally understood as a platform that provides access to Internet services, such as e-mail and Web browsing, etc. From a technical point of view, however, the primary meaning of the term Internet access should be understood as a physical and logical access to the core of the network, including all functionalities needed to enable the user to establish a connection to further entities in the Internet and to run advanced services [8].

Providers often advertise the maximum values of the throughput, which is rarely accessible, due to it being strongly connected strongly connected with the variable traffic load and the still increasing demand for data transmission bandwidth in recent years. Many users often expect such throughputs for most of the day, irrespective of the time and network conditions.

Unfortunately, according to the CISCO forecast, presented in Visual Networking Index [9], global IP traffic will increase nearly threefold over the next 5 years and by 2020 will reach 2.3 ZB per year.

Moreover, traffic load varies significantly during the day. Busy-hour (the busiest 60-minute period in a day) Internet traffic is growing more rapidly than average Internet traffic. It increased by 51 percent in 2015, compared with a 29-percent growth in average traffic. It means that service providers will face even higher network load fluctuations and more serious traffic engineering problems than up to now.

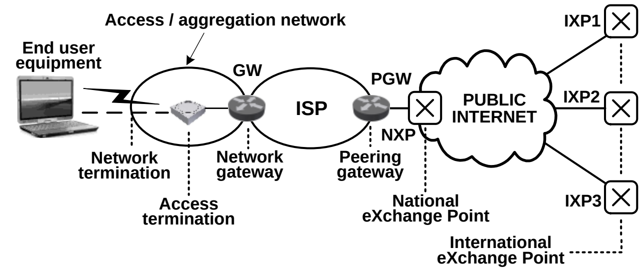

Figure 1. General overview of elements and network sections of IAS.

Users can be connected to the various ISPs via the access networks, using wired or wireless connections. Communication over the Internet requires data interchange over different National and International eXchange Points (NXPs and IXPs). Figure 1 presents a generic overview of the elements, network sections and interfaces of the IAS according to [10].

A very important issue is the proper definition of the Internet Access Service (IAS). The answer to this question is not only crucial for the users, who are usually not familiar with the technical details, but also for the providers as well, because it determines the user-to-network and network-to-network interfaces and also the responsibilities of the providers.

Finally, it says how IAS quality should be measured and how the results can be interpreted and compared between different providers and their end-users. It is especially important in the light of European regulation [11] on the rights of users to be informed about the quality of their services.

III. IAS MEASUREMENTS

Identifying the parameters that may affect the Quality of Service (QoS), locating the points at which the measurements should be performed and specifying the measurement scenarios is a sequence that should be done before the measurements. Simply speaking, one should specify "what, where and how" should be measured to provide ISPs and users with a thorough knowledge of the QoS.

The measurements fall into two groups: so called "In-net" and "Over-The-Top" (OTT). The first case covers the ISP's area - the area on which it acts. European Consumer Center (ECC) Report [10] specifies a list of technical quality parameters proposed to be measured during a technical evaluation of IAS.

Many National Regulatory Authorities (NRAs) or other national institutions agree that the list is too long. They also consider it to be too complicated and incomprehensible to the average user. Thus, they propose the selection of a subset of parameters. After consulting an abundance of documents [8][10][12] and different points of view, the ECC has proposed a list of minimum technical parameters that take their influence on the most popular Internet applications into account. Table 1, based on [10], illustrates popular services and the relevance of the network performance parameters to the performance or quality of those services. The relevance ranges from "−" (irrelevant) to "+++" (very relevant). The following quality metrics have been selected: data transmission rate, delay, delay variation, packet loss ratio, and packet error ratio.

TABLE I. RELEVANCE OF NETWORK IMPAIRMENT PARAMETERS TO VARIOUS APPLICATIONS

Service	*Data transmission speed*		*Delay*	*Delay variation*	*Packet loss*	*Packet error*
	Down	*Up*				
Browse (text)	++	−	++	−	+++	+++
Browse (media)	+++	−	++	+	+++	+++
Download file	+++	−	+	−	+++	+++
Transactions	−	−	++	−	+++	+++
Streaming media	+++	−	+	−	+	+
VoIP	+	+	+++	+++	+	+
Gaming	+	+	+++	++	+++	+++

The data transmission rate is probably the most relevant parameter, nearly mentioned in every ISP's offer. It is defined as the data transmission rate that is achieved separately for downloading and uploading specified test files between a remote website and a user's terminal equipment [8]. The next parameter is delay, defined as half the time (in ms) that is needed for an ICMP packet to reach a valid IP address. This parameter also has a significant influence on many applications available over the Internet and is already being used by many NRAs, operators and Web-based speed meters. There are also some applications that are very sensitive to delay variation and this parameter is therefore selected for measurements. The exact definition of delay variation can be found in [12][13].

IP packets can sometimes be dropped, e.g., due to a small buffer size of the network nodes or poor (radio) connection, even if the transmission rate, delay, and delay variation remain good enough. Such packet loss can significantly affect all data-based applications. Moreover, UDP-based applications, such as Voice over IP may also not work properly in such conditions. Packet loss ratio can be defined as the ratio of the total lost IP packet occurrences to the total number of packets in the population under examination [13]. The parameter that may have an influence on the quality of service is the packet error rate and was therefore also included in the basic set of measured parameters shown in Table 1. The IP packet error ratio is sometimes called the packet error ratio and is defined as the ratio of the total faulty IP packet occurrences to the total number of successful IP packet deliveries plus the faulty IP packet occurrences within a population of interest.

Internet access is no longer provided by a single network or service provider, as was the case with traditional voice communication in Public Switched Telephone Networks (PSTNs). Nowadays, a user gains an indirect access to the public Internet, as shown in Figure 1. Therefore, the overall quality of services (or, in general, Internet access) is a combination of the performance of all the elements involved in the connection.

Different approaches to QoS measurements are discussed in literature. One of the classifications points out the methods as follows:

- carried out by the carefully selected users running the measurement tests from designated locations (or users' homes) and using special purpose equipment [10][14][15],
- large-scale user-driven tests, performed by software agents installed on PCs, tablets, smartphones, etc. [14].

On the other hand, the measurements can be performed by network or service providers, regulators or designated third-party institutions. Different solutions are used in different countries. Many providers do it individually but their results may be regarded by users as non-objective. Thus, external institutions are needed here. Such institutions are very often national regulators or the external companies hired by the regulators. The first solution is used, e.g., in Portugal [14], while the second approach, based on "QoS Memorandum" [16], is used in Poland.

At the European level, the minimum set of QoS parameters and measurement methods for retail Internet Access Service has been described in [10]. According to this, the measuring points to be used during the IAS quality assessment may be specified (Figure 2).

Three evaluation methods (scenarios) are relevant to the measurements connected with IAS quality assessment. The methods encompass an examination of the access network, the ISP network and the network connections to NXP or IXP.

Their names are listed below:

- QoS evaluation within the ISP leg,
- QoS evaluation between the Network Termination Point (NTP) and NXP(s),
- QoS evaluation between the NTP and IXP(s).

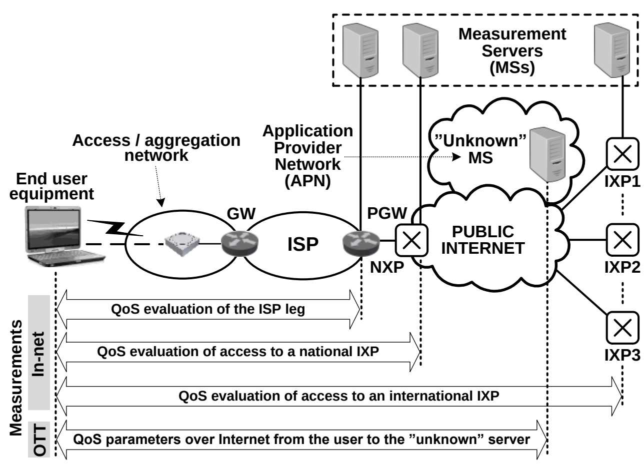

Figure 2. Internet Access Service quality assessment.

Depending on the scenario, the measurement server should be located in the right place (cf. Figure 2).

In order to only test the access network, the test server should be located as close as possible to the gateway (GW) between the access network and the ISP network. In the case of evaluating the entire ISP leg quality, the test server should be placed near the public Internet interface (PGW in Figure 2). Locating the test server in the National eXchange Point (NXP) allows the network performance parameters of different ISPs to be compared. The quality results achieved in this scenario seem to be far closer to the quality of Internet connection, as perceived by users, than the results in the "ISP leg" scenario.

It can be seen that the Internet Access Service quality assessment is therefore a very demanding issue, especially as users care about their own quality experience, which is commonly understood as unrestricted, high-quality and having a reliable access to the applications they use and the content they seek out online. This is the reason for performing the second type of measurements presented in Figure 2. They were called "OTT measurements", because they allow the performance parameters of specific applications run by the users to be tested and thus they, in general, better reflect the quality of service as perceived by the user. Nonetheless, these are measurements of the objective parameters and, in the next step, should be transformed into the quality measures as perceived by users. Mapping the measured QoS factors to the QoE ones is often quite a complicated process. The next paragraph presents an example of WWW service quality assessment as perceived by users.

IV. SERVICE QUALITY PERCEIVED BY USERS

In this paragraph, we present an example of the service quality assessment procedure based on the WWW service. The WWW is one of the most popular services, if not the most important of all, used by Internet users. Many of them assess the Internet quality through the lens of Web browsing and information searching on the Internet. The main parameter that influences the service quality, as perceived by the user, is Web page opening (loading) time. In other words, the end-to-end (e2e) delay between the user's request and the time when the page is open on the user's display is the most important. The WWW service quality evaluation procedure will be treated as one of the factors that influence the user's perception of the IAS. The WWW service evaluation in the real network may be performed as shown in Figure 3.

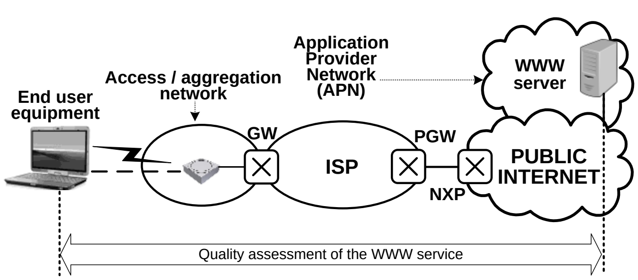

Figure 3. WWW service quality assessment.

After objective measuring of the Web page opening times, the service quality perceived by the end-users, i.e., the relation between QoS and QoE, should be found. In other words the QoE model for the service should be determined. By presenting the WWW quality assessment, the author would like to underline that measuring and presenting only the network performance parameters to the customers, discussed in previous sections of the paper, may not be sufficient for determining the IAS quality as perceived by the users. There is a need to check the service quality experienced by them and building such a model requires a special laboratory environment. The one used here is presented in Figure 4.

The laboratory test-bed consists of a WWW client with a measuring tool, a test server that hosts a set of special prepared WWW pages and the Network Emulator (NE). All the machines and software run under the MS Windows operating system. As a user client, the Mozilla Firefox browser was used while the measuring tool was the Wireshark protocol analyzer. The NE was capable of emulating the impairment parameters such as network delay, jitter and packet loss. This stage of the measurements only studied the impact of the delay on the service quality as perceived by the users. The delays were randomly generated by the NE while the users tried to open the Web pages on the test server. Next, the packets were captured by the Wireshark and analyzed. The users did not know the strict values of the delays, but they did see the effects and tried to assess them.

It was clear that the Web page opening times had a decisive influence on QoS values for the WWW service. It was to be expected that increases in end-to-end delay would lead not only to deterioration of QoS but to QoE values as well. Quality of Experience was expressed by the user's evaluation grades according to the Mean Opinion Score (MOS) scale [17]. The first observations confirmed these expectations, but it was also noticed that the subjective opinion of users depended highly on the page properties, i.e., their content, layout, construction (static, dynamic), etc. For subjective measurements the WWW reference page was needed. Static Web pages were launched on the test server and the contents of these pages were different. One of them was prepared according to ETSI reference page requirements [18], while in the second case a photo gallery was used. In that case a special scenario of the WWW pages presentation and evaluation was prepared.

The scenario assumed that every user, when evaluating Web opening times (equivalents of end-to-end delays during normal Web browser use), should give his grade after seeing several photos so that he would be better able to make a judgment.

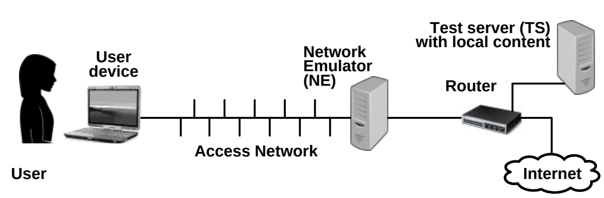

Figure 4. The laboratory test-bed for the WWW QoE assessment.

The test was performed on a user's PC (WWW client with a measurement tool). Additionally, Wireshark software installed on the client's PC (as a second tool) was used to capture IP packet streams and to register the end-to-end delay time. This was defined as the difference between the point in time at which the Web page was requested and the point in time at which all data needed for the display of the Web page were received. The end-to-end delay was varied throughout the course of the experiment using the NE. It was noticed that the Web page opening times that were registered at the user site played a crucial role in the subjective evaluation of WWW service quality (QoE). There were several groups of professional users (each group of 10) taking part in the experiment (more than 70 users in total). They gave their subjective grades for WWW service quality in a range from 1 to 5 on the MOS scale. More than 1500 test measurements were conducted. In the next step the statistical analysis has been performed.

V. THE QOE MODEL

The measurements show that the grades of users are inversely proportionate to the Web page opening times. To speak in more detail, the people who took part in the evaluation test were quite critical with regards to the service under analysis: a rapid decrease in the quality can be observed for the Web page opening times (T) covered in the first few seconds. It shows that users are very critical in their opinions and do not accept long delays. The longer the Web opening times, the lower grades users give. For the delays exceeding 10 s, the grades of users tend to be significantly lower at a level of 2, which means that such long times are unacceptable for WWW users.

The analysis of the results leads to the conclusion that users had a considerable problem with evaluating Web page opening times with very high fluctuations. The measurement results obtained are consistent with those presented in literature [19]. It can be noticed that users are willing to award very high grades for the service (MOS = 5) when opening times are under 2 s, while the lower grades (MOS less than 2) are given when opening times are 8 s and more. In individual cases the evaluation grades may differ significantly from the majority of the scores and thorough statistical analysis should therefore should be carried out. As can be seen in Figure 5, the mean values for the specific page opening times were not only determined, but min and max values and standard deviation as well.

The correlation between the opening times and the user grades achieved here is at a level of 80 %. The standard deviation is indicated by the dashed boxes in Figure 5, while whiskers represent the distances between the minimal and maximal values of the captured page opening times. This shows a high level of user uncertainty during the evaluation process.

As it is known from the former experiments [19][20], during long waiting times many users begin to consider whether waiting for the page to open makes sense, and many of them resign. To find a precise relation between the captured values of Web opening times and the quality experienced by users, a regression model was used.

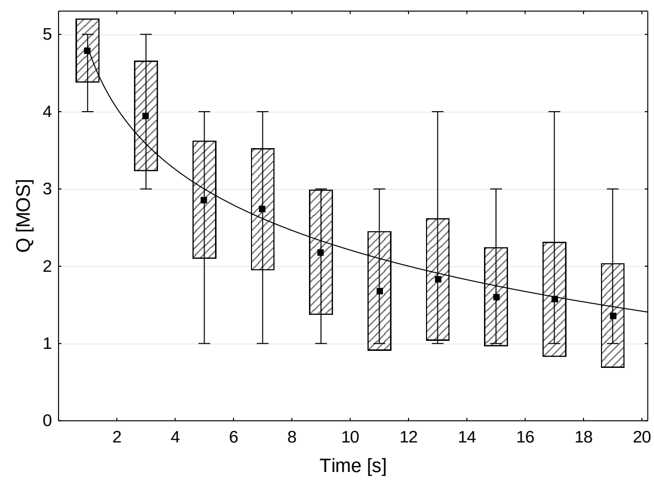

Figure 5. Subjective evaluation of WWW page opening times in MOS scale.

The model we derived can be described by the following formula:

$$Q = 4.84 - 2.63 \log_{10} T, \quad (1)$$

where: T is the Web page opening time.

The logarithmic line (Figure 5) represents the Q value (in MOS scale) as a function of Web opening times. The statistical analysis proved that the model fits the data very well, with the coefficient of determination (R2) above 0.9. It means that the obtained outcomes are replicated by the model in at least 90 % of the time. Confirmation of such a user's QoE distribution can be found in the analysis results presented by the above-mentioned ITU-T recommendation [19], where attention had also been drawn to the logarithmic nature of the relation between QoS and QoE in such a case.

A possibility of determining the prospective MOS value by managing the opening times is very valuable and more convenient for the provider than performing the subjective evaluations, which are time consuming and more expensive.

VI. CONCLUSION AND FUTURE WORK

Internet Access Service is a key factor that influences a user's perception of all the services provided on the Web. Thus, service providers have to do all their best to offer a good quality IAS. Moreover, they should monitor the network transmission parameters and be up to date with their values. Usage of the appropriate measurement methods is therefore very important. The methods can use different scenarios. In order to make the results credible and comparable with others, these scenarios should be clear and measurement interfaces and procedures have to be clearly defined. The paper shows the different measurement solutions that can be used. In the second part of the paper the author stressed the importance of subjective quality assessment methods, which are based on the experience of users and give more information about their perception. They assess the Internet Access quality through the quality of the services that they use. One of the most popular is the WWW service. Therefore, the author presented the example of a Web browsing quality evaluation scenario, specified the key quality parameter and showed the results of measurements. At the end, the QoE model was proposed and discussed. The main conclusion is that the quality measurements should not only take into account the objective parameters, but subjective parameters as well. Obviously, the set of the parameters depends on the service. Future work will be devoted to WWW QoE model enhancement by specifying a wider set of parameters to be measured and to also build reference Web pages that will be more representative for current Internet content.

Acknowledgment

The author would like to thank Pawel Bardowski for his contribution to the test-bed set-up and to Zbigniew Salamacha for his help during the editorial process.

References

[1] European Commission. *EUROPE 2020 - A strategy for smart, sustainable and inclusive growth*. [Online]. Available from: https://ec.europa.eu/digital-single-market/en/europe-2020-strategy, [retrieved: 12, 2016].

[2] European Commission. *A Digital Agenda for Europe*. [Online]. Available from: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52010DC0245&from=EN, [retrieved: 12, 2016].

[3] European Commission. *Connectivity for a Competitive Digital Single Market - Towards a European Gigabit Society*. [Online]. Available from: http://ec.europa.eu/newsroom/dae/document.cfm?doc_id=17182, [retrieved: 12, 2016].

[4] BEREC. *How do consumers value Net Neutrality in an evolving Internet marketplace?* [Online]. Available from: http://www.berec.europa.eu/eng/netneutrality/, [retrieved: 12, 2016].

[5] Official Journal of the European Union. *Regulation (EU) 2015/2120 of the European Parliament and of the Council of 25 November 2015*. [Online]. Available from: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32015R2120&from=en, [retrieved: 12, 2016].

[6] BEREC. *Guidelines on Transparency*. [Online]. Available from: https://ec.europa.eu/digital-single-market/en/news/berec-guidelines-transparency-scope-net-neutrality-best-practices-and-recommended-approaches, [retrieved: 12, 2016].

[7] BEREC. *The value of Network Neutrality to European Consumers*. [Online]. Available from: http://www.berec.europa.eu/eng/netneutrality/, [retrieved: 12, 2016].

[8] ETSI. *EG 202 057-4. Speech Processing, Transmission and Quality Aspects (STQ); User related QoS parameter definitions and measurements; Part 4: Internet access*. [Online]. Available from: http://www.etsi.org/deliver/etsi_eg/202000_202099/20205704/01.02.01_60/eg_20205704v010201p.pdf, [retrieved: 12, 2016].

[9] CISCO. *White paper: Cisco VNI Forecast and Methodology, 2015-2020, June 2016*. [Online]. Available from: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html, [retrieved: 12, 2016].

[10] ECC Report 195. *Minimum Set of Quality of Service Parameters and Measurement Methods for Retail Internet Access Services, April2013*. [Online]. Available from: http://www.erodocdb.dk/Docs/doc98/official/pdf/ECCREP195.PDF, [retrieved: 12, 2016].

[11] Official Journal of the European Union. *DIRECTIVE 2009/136/EC of the European Parliament and of the Council, 337/11, 25 Nov. 2009*. [Online]. Available from: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32009L0136, [retrieved: 12, 2016].

[12] ITU-T. *Rec. Y.1541: Network performance objectives for IP-based services.* [Online]. Available from: https://www.itu.int/rec/T-REC-Y.1541/en, [retrieved: 12, 2016].

[13] ITU-T. *Rec. Y.1540: Internet protocol data communication service - IP packet transfer and availability performance parameters*. [Online]. Available from: https://www.itu.int/rec/T-REC-Y.1540/en, [retrieved: 12, 2016].

[14] R. Nunes, R. Pereira, Rui Valadas, and S. Parranc, “Agent-Based Platform for Continuous Measurement of Internet Access Quality of Service” 16th International Telecommunications Network Strategy and Planning Symposium (Networks), Funchal, 2014, pp. 1-6, doi: 10.1109/NETWKS.2014.6959251.

[15] SamKnows. *The global platform for internet measurement*. [Online]. https://www.samknows.com, [retrieved: 12, 2016].

[16] UKE. *Memorandum QoS report* (in Polish). [Online]. Available from: https://www.uke.gov.pl/raport-koncowy-memorandum-jakosci-13373, [retrieved: 12, 2016].

[17] ITU-T. *Rec. P.800.1: Mean opinion score (MOS) terminology*. [Online]. Available from: https://www.itu.int/rec/T-REC-P.800.1/en, [retrieved: 12, 2016].

[18] ETSI. *Kepler reference web page*. [Online]. Available: http ://docbox.etsi.org/STQ/Open/Kepler/, [retrieved: 12, 2016].

[19] ITU-T. *Rec. G.1030: Estimating end-to-end performance in IP networks for data applications*. [Online]. Available from: https://www.itu.int/rec/T-REC-G.1030/en, [retrieved: 12, 2016].

[20] P. Bardowski, J. Klink, and T. Uhl, “New metrics for WWW service quality assessment,” Przegląd Telekomunikacyjny (in Polish), No. 8-9, 2013, pp. 644-649.

Impact of Network and Service Paradigm Shift on Evolution of the QoE and QoS Concepts

Jacek Oko
Wrocław University of Science and Technology
Telecommunications and Teleinformatics Department
Wrocław, Poland
e-mail: jacek.oko@pwr.edu.pl

Marek Średniawa
Warsaw University of Technology
Institute of Telecommunications
Warsaw, Poland
e-mail: mareks@tele.pw.edu.pl

***Abstract*—We are witnessing a number of paradigm shifts in many Information and Communication Technology (ICT) areas. 5G, Future Internet and very high resolution digital TV defined a new vision of services with novel approaches to their implementation, deployment and operation based on concepts of Software Defined Network (SDN) and Network Function Virtualization (NFV). Increasing transmission capabilities in both mobile and fixed networks became an enabler for a new generation of multimedia services and applications. Evolution of digital video technology and competition of operator supported services with Over The Top (OTT) applications caused change in understanding of Quality of Service (QoS) and Quality of Experience (QoE) concepts. This paper surveys the change of QoE approaches and interpretation in the context of new services and applications.**

Keywords-QoE; QoS; multimedia services.

I. INTRODUCTION

Delivery of services of appropriate quality has always been one of the key goals of telecom operators and service providers. Evolution of services from fixed network plain voice and data towards rich and complex mobile multimedia and Internet based OTT (Over The Top) applications caused a significant change in quality concepts and approaches to their provision, measurement, monitoring and management.

For a long time, a network centric approach based on the notion of QoS defined by strictly technical parameters associated with data transmission among service access points (ITU-T Rec. X.200) like: transmission delay, jitter, throughput, bit error rate, probability of loss or duplication of data, has dominated. This "engineering" perspective neglected users' point of view. There is of course an intuitive relationship between QoS and quality of service observed by a user. On the one hand, meeting a set of QoS criteria might not guarantee end-user satisfaction. On the other hand, a QoS problem, e.g., higher level of delay or jitter may not affect quality of voice or video to a level that causes users complaints. However, a QoE problem, e.g., macroblocking in a video stream, may be affected by QoS problems like e.g. dropped IP packets leading to continuity count errors. In order to represent also the users' centric perspective, a concept of QoE was introduced. It focuses on the perceptual quality of services from the users' point of view.

According to ITU-T Rec. G.100 [2], QoE is defined as *the overall acceptability of an application or service, as perceived subjectively by the end-user*. An alternative but complementary definition was proposed by the Qualinet "White Paper on Definitions of Quality of Experience" [14]. QoE is defined there as: *the degree of delight or annoyance of the user of an application or service. It results from the fulfilment of his or her expectations with respect to the utility and/or enjoyment of the application or service in the light of the user's personality and current state*.

It is a metric of how well a network satisfies the end-user's requirements and expectations. QoE establishes an aggregated view by covering the complete end-to-end system (client, terminal, network, services infrastructure, etc.) and its effects. QoE is also influenced by user's internal state (e.g., expectations), the characteristics of the designed system (e.g., capabilities) and the context or the environment within which the interaction occurs. In research papers on QoE, many classifications of the QoE Influencing Factors (IF) were proposed. For example, in Qualinet project [14], three categories of QoE influence factors were identified:

1. human (e.g., gender, age, education);
2. system (e.g., bandwidth, security, resolution);
3. context (e.g., location, movements, costs).

Figure 1 summarizes QoE IF considered in twenty selected publications described in [10].

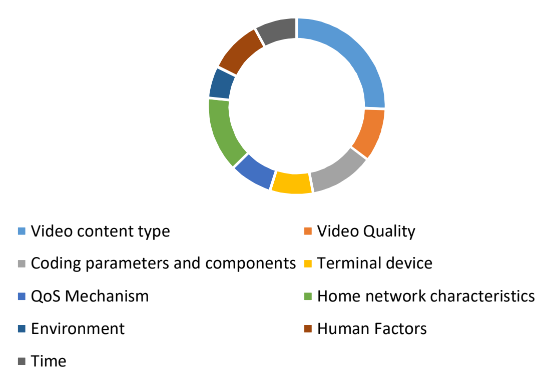

Figure 1. Factors influencing video QoE [10].

 ISBN: 978-1-61208-528-9

It should be noted that QoE, in contrast to QoS, is subjective in nature and involves context and potentially specific user expectations of "soft" nature, which might not be directly expressed as values of parameters.

In theory, the QoE for a given application is a function of the network QoS parameters, but its discovery might be difficult due to complexity of a service delivery environment. Such perspective makes QoE a challenging issue and calls for individual treatment of each category of service together with its delivery infrastructure. These new areas emerged due to developments in networking: 5G, Future Internet, media: High Definition (HD) voice, Ultra- and Super Ultra HD (UHD/SUHD) video, services and applications: IP TV, VoD, YouTube, Vimeo, Netflix, mobile gaming, and paradigm shift taking advantage of new concepts and approaches such as SDN, NFV, Cloud Computing (CC) and Self Organizing Networks (SON).

The paper discusses the change of QoE approaches and interpretations in context of new services and applications.

II. QoS/QoE in 4G

The evolution of mobile network architecture from 3G to 4G featured significant change in QoS mechanisms; however, they were still a starting point for mapping to QoE. One of the crucial changes is also common handling of voice and data in an all IP packet network [3].

The UMTS 3G [12] defines four QoS classes differing by traffic delay sensitivity: *Conversational*, *Streaming*, *Interactive* and *Background*. Conversational and Streaming classes were dedicated for real-time services like voice and video streaming, while the other two Interactive and Background were appropriate for best-effort services with less stringent delay requirements (e.g. web browsing and progressive download of video content). The UMTS High Speed Downlink Packet Access (HSDPA) standard [20] introduced an additional QoS control means - a Scheduling Priority Indicator (SPI), a number from the 0-15 range, which could be used by each nodeB for scheduling. Due to their local character, SPIs priorities had limited applicability from the core network perspective.

Development of 4G Long Term Evolution (LTE) standard [21] assumed delivery of real-time, multimedia services with support for appropriate levels of QoE [4]. It introduced a QoS-aware mechanism for end-to-end service delivery based on Evolved Packet System (EPS) bearers and Quality of Service Class Identifiers (QCIs). The end-to-end QoS is established from User Equipment (UE) to the Packet Data Network Gateway (PDN-GW) in a core network using bearer service which provides the QoS level of granularity for different service flows. EPS bearer QoS profiles include following parameters: QCI, Allocation and Retention Priority (ARP), Guaranteed Bit Rate (GBR), and Maximum Bit Rate (MBR). The QCI is a number from the 1-8 range used to control bearer level packet handling in scheduling and admission control. Two categories of QCIs are defined:

- Guaranteed Bit Rate (GBR) for applications such as conversational voice, video streaming, buffered streaming and real time gaming, with QoS constraints to be observed;
- Non-Guaranteed Bit Rate (Non-GBR) for applications such as voice, video, TCP based applications and IP Multimedia Subsystem (IMS) signaling, with no QoS constraints.

The 4G LTE bearer centric QoS architecture has a constraint. It is not able to differentiate traffic flows that are served by the same EPS bearer unless dedicated EPS bearers are used for different traffic flows. It means that different traffic flows of the same service which needs better level of QoE/QoS may not differ in 5-tuple (source IP address, destination IP address, source port, destination port, protocol), identifying packets associated with a related unidirectional flow [12]. As a consequence, though there is rich support for QoS LTE there still a need for extra resources to deliver an intended QoE for real-time applications. In case of video services using GBR or QoS to give priority for forwarding of video related packets will improve QoE but at the same time will deteriorate quality of best effort data traffic. Moreover, diverse user devices with different screen sizes and resolutions, i.e., tablets and smartphones, but in the same radio conditions, should be treated individually.

III. QoS/QoE in 5G

Forthcoming 5G network makes a great step forward in comparison to 4G, in terms of speed (1-10 Gbps), extremely low latency (1 ms round trip time), base station capacity, longer battery life, availability (99.999%) and perceived quality of service (QoS). All these features will also contribute to improvement of QoE. Dominating role of multimedia traffic with growing share of HD and UHD video [1] will broaden a list of factors to be considered in the context of QoE. This in turn will make QoS to QoE mapping even more complex. There is also another dimension of complexity for QoS and QoE. Implementation of 5G networks will be based on the concepts of SDN and NFV and will employ cloud computing technology.

In 5G, services may use different allocation of network functions in Radio Access Network (RAN) and Core Network (CN) both for user (u-) and control (c-) plane according to required QoS/QoE needs. This capability and decoupling of control and data plane in SDN provides flexibility alleviating improvement of QoE [15][16].

In 5G NORMA project [19], SDN technology is used to implement a prototype of virtualized QoS/QoE mapping functions using open interfaces and flexible selection of functional blocks to integrate specific requirements for each user. Such solution will provide network programmability capabilities allowing third parties, e.g., virtual operators to set-up their specific QoS/QoE control strategies. It will enable support of fully dynamic, context aware QoE/QoS management capable to dynamically set QoS/QoE target based on detected application flows of the end users and adapt the end-to-end resource allocation and the data plane functions accordingly.

CC technology used for 5G architecture will also enable applying of Mobile Edge Computing (MEC) concept to implement QoE by making caching/replacement decisions

 ISBN: 978-1-61208-528-9

based on both content context (e.g., segment popularity) and network context (e.g., RAN downlink throughput).

IV. RELATIONSHIP BETWEEN QOS AND QOE

A generic relationship between QoS and QoE defines three zones (Figure 2) and is described by (1). For QoS disturbance from Zone 1, QoE has a high value, representing the fact that user's appreciation is not affected. When the QoS deteriorates and disturbance reaches Zone 2 the QoE decreases. Finally, when the QoS disturbance grows and reaches Zone 3, the QoE may cause user abandons a service because of its unsatisfactory quality. Generally, when the QoS disturbance parameter increases, the QoE metric and user's perception of quality decrease [6]. Getting insight into the nature of the relationship between QoS and QoE, and understanding it to a level enabling control over QoE is still a research topic.

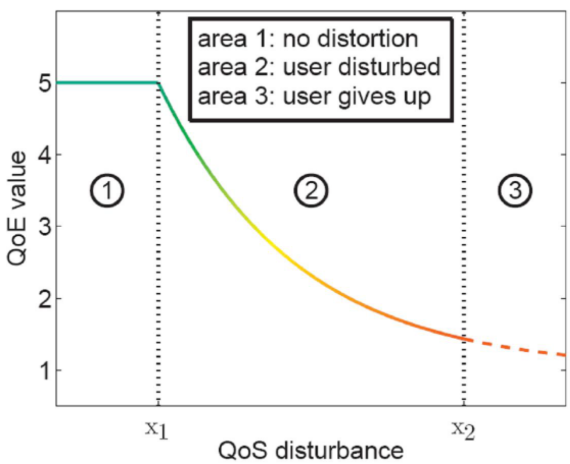

Figure 2. Generic mapping curve between QoS and QoE [6].

$$QoE = \alpha \exp(\beta - QoS) + \gamma \quad (1)$$

where α, β, γ, x1 and x2 are configurable positive value parameters. Usually QoS is calculated by taking into consideration four generic criteria: availability, reliability, delay and capacity. These four criteria are independent of context and in practice are represented by parameters relevant and specific for a case (e.g. round trip delay, jitter, packet loss rate, bit error rate, throughput, bandwidth, etc.). The above formula represents an objective approach to QoE known as "IQX hypothesis" which is typically used to estimate the QoE for VoIP services and web browsing [6]. It expresses QoE as an exponential function of the QoS degradation. Its weaknesses are:

- Doubts concerning values of parameters which might be relevant only for specific use cases;
- Neglecting the actual end users' opinions.

Their perspective is taken into consideration by subjective metrics of QoE, which are based on psychoacoustic/visual experiments with human users. Historically, the most important and the oldest metric for assessment of perceived service quality or human perception is the Mean Opinion Score (MOS) with a five quality levels (from 5=excellent to 1=bad) Absolute Category Rating (ACR) defined by ITU-T Rec. P.800. Despite criticism for being too simplistic, it is still very popular not only for voice but also for multimedia services (e.g., UHD TV) [5]. The drawbacks of MOS method are complexity and costs.

Evolution of video quality assessment methods and issues of QoS/QoE mapping are discussed in detail in [4][8]. Table I presents video QoE standards developed by ITU-T and Video Quality Experts Group (VQEG) [22].

TABLE I. VIDEO QOE STANDARDS [22][8]

Image res..	*Subjective estimation*	*Full reference*	*Non Reference*		*Reduced Reference*
SDTV HDTV	ITU-R: BT.500 ITU-T: J.140, J.245	ITU-T: J.144 J.341	ITU-T: P.1201, P.1202, G.1071	VQEG: RRNR-TV, HDTV	ITU-T: J.147, J.249 J.342
VGA CIF QCIF	ITU-T: P.910, P.911	ITU-T: J.247		VQEG: MM Project	ITU-T: J.246

Subjective methods have been studied for many years and have enabled researchers and operators to get better understanding of the subjective aspects of QoE. Typically, the results of subjective experiment are quality ratings obtained from users during use of the service (in-service) or after service use (out-of-service), which are then averaged into MOSs and extended to other ITU standard-based subjective assessment procedures classified by type of application and media (Table I).

Several models were proposed for QoS/QoE mapping [7]-[11]. A comprehensive discussion can be found in [13] and [18].

V. CONCLUSIONS

The relationship between QoS and QoE is a challenging task and a moving target due to: evolution of services towards collaborative multimedia, migration of mobile network architecture to a new 5G networking paradigm based on virtualization, and extension of the list of QoE influencing factors. Managing and controlling QoE is still a research topic.

CC technology applied in 5G architecture enables usage of Mobile Edge Computing (MEC) concept to implement QoE by making caching/replacement decisions based on both content context (e.g., segment popularity) and network context (e.g., RAN downlink throughput). For this case, mobility can be used as context information for future location and anticipated trajectory enabling pre-caching of content-based on user location.

Ubiquity of smartphones and tablets together with increasing performance of networks open an opportunity for substantial reduction of costs of conducting MOS tests by

 ISBN: 978-1-61208-528-9

using crowdsourcing for subjective assessments of QoE of multimedia services and applications.

Possibly new performance metrics are needed for more appropriate evaluation of QoE/QoS of mobile web services. Diversity of terminals that differ with resolution of screens and other features like power consumption or supported data rates requires separate treatment in terms of QoE. Common performance parameters like speed in bits per second should be replaced by screen per second in order to express webpage delivery and display. Tests conducted by operators revealed that there is a gap between measured QoS indicators and user perceived experience. Large difference in network performance might result in thin margins of users' experience as a result of complex relationship between QoS and QoE influencing factors.

For video, a natural challenge will be providing satisfactory QoE for UHD TV content up scaled from UHD and HD formats.

Changing communication landscape calls also for redefinition of traditional Service Level Agreements (SLAs) which are closely related to both QoS and QoE. Today, QoE models could be used as guides for setting and negotiating proper SLAs. By analogy to the mapping between QoS and QoE, it is worthwhile to consider an idea of Experience Level Agreement (ELA) as a special type of SLA which establishes a common understanding of the quality levels that the customer will experience through the use of the service [17].

REFERENCES

[1] "Cisco visual networking index: Forecast and methodology, 2015-20120, 2016".

[2] ITU-T P.10/G.100, "Vocabulary and Effects of Transmission Parameters on Customer Opinion of Transmission Quality," 2008.

[3] D. Soldani, M. Li, R. Cuny, "QoS and QoE Management in UMTS Cellular Systems", Wiley, 2007.

[4] C. Wen Chen, T. Dagiuklas, P. Chatzimisios, L. Atzori, "Multimedia Quality of Experience (QoE): Current Status and Future Requirements", Wiley, 2016.

[5] A. Takahashi, D. Hands and V. Barriac, "Standardization activities in the ITU for a QoE assessment of IPTV," IEEE Communications Magazine, vol. 46, no. 2, pp. 78-84, 2008.

[6] M. Fiedler, T. Hossfeld and P. Tran-Gia, "A Generic Quantitative Relationship between Quality of Experience and Quality of Service," IEEE Network Magazine, vol. 24, no. 2, pp. 36-41, 2010.

[7] P. Reichl, B. Tuffin and R. Schatz, "Logarithmic laws in service quality perception: where microeconomics meets psychophysics and quality of experience," Telecommunications Systems Journal, vol. 5, no. 2, pp. 587-600, 2011.

[8] Y. Chen, K. Wu, Q. Zhang, "From QoS to QoE: A Tutorial on Video Quality Assessment", IEEE Communications Surveys & Tutorials, Vol.17(2), pp.1126-1165, 2015.

[9] H. J. Kim, S. G. Choi, "A study on a QoS/QoE correlation model for QoE evaluation on IPTV service", Advanced Communication Technology (ICACT), The 12th International Conference on, vol. 2, pp. 1377-1382, 2010.

[10] M. Alreshoodi, J. Woods, "Survey on QoE/QoS Correlation Models For Multimedia Services," International Journal of Distributed and Parallel Systems (IJDPS), vol. 4, no. 3, pp. 53-72, 2013.

[11] V. Chervenets, V. Romanchuk, H. Beshley, A. Khudyy, "QoS/QoE correlation modified model for QoE evaluation on video service", 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), pp. 664-666, 2016.

[12] ETSI TS 123 107, Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; "Quality of Service (QoS) concept and architecture" (3GPP TS 23.107 version 11.0.0 Release 11), ETSI Technical Specification, 2012.

[13] P. Brooks, B. Hestnes, "User measures of quality of experience: why being objective and quantitative is important", IEEE Network, vol.24, Issue 2, pp. 8-13, 2010.

[14] P. Le Callet, S. Möller, A. Perkis, "Qualinet White Paper on Definitions of Quality of Experience (2012)", European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003), Lausanne, Switzerland, Version 1.2, 2013.

[15] M.S. Mushtaq, S. Fowler, B. Augustin, A. Mellouk, "QoE in 5G Cloud Networks using Multimedia Services", IEEE Wireless Communication and Networking Conference (WCNC'16), 2016.

[16] E. Liotou, H. Elshaer, R. Schatz, R. Irmer, M. Dohler, N. Passas, L. Merakos, "Shaping QoE in the 5G ecosystem, 7th International Workshop on Quality of Multimedia Experience, pp.1-6, 2015.

[17] P. Reichl, P. Zwickl, A. Passarella, A. Sackl, "Economics of QoE in the Future Internet of People (IoP)", ETSI STQ Workshop Vienna, 2015.

[18] S. Baraković, L. Skorin-Kapov, "Survey and Challenges of QoE Management Issues in Wireless Networks" Journal of Computer Networks and Communications, special issue on "Improving QoS and QoE for Mobile Communications", 2013.

[19] "Deliverable D5.1. Definition of connectivity and QoE/QoS management mechanisms – intermediate report", 5G Novel Radio Multiservice adaptive network Architecture (5G NORMA) Project, 2016.

[20] 3GPP TS 25.308 "UTRA High Speed Downlink Packet Access (HSDPA); Overall description;Stage 2", 2001.

[21] 3GPP, "Overview of 3GPP Release 8 V0.3.3", 2014.

[22] VQEG, https://www.its.bldrdoc.gov/vqeg/vqeg-home.aspx.

Resource Description for Additive Manufacturing – Supporting Scheduling and Provisioning

Felix W. Baumann, Julian R. Eichhoff, and Dieter Roller
Institute of Computer-aided Product Development Systems
University of Stuttgart
Email: firstname.lastname@informatik.uni-stuttgart.de

***Abstract*—For an enhanced automated usage of 3D-printers in case of multiple available 3D-printers, such as in Cloud Manufacturing or Cloud Printing services, the requirement arises to select and provision suitable resources for user provided model files. As Additive Manufacturing (AM) consists of a number of different technologies, ranging from fabrication using thermoplastic extrusion to electron beam based curing of metal powder, the necessity is evident to enable users to describe limitations, capabilities, interfaces and requirements for a these resources in a machine readable and processable format. This resource description enables the discovery and provisioning of appropriate resources within a service composition, where 3D-printing resources are regarded as manufacturing services themselves. In order to compose a service from these hardware resources, the comprehensive description of such resources must be provided. With this work, we provide an abstract and universal capability description framework of such 3D-printing resources. The framework consists of an ontology for the resources of the AM Domain, a flexible XML schema and the implementation in a cloud-based 3D-printing system. With this resource description both hard- and software resources are universally defined. Applied to systems with multiple 3D-printers, a scheduling component is capable of resource discovery. This selection is based on the matching of described capabilities, status information and derived requirements from specific 3D-printing job definitions.**

Keywords*—3D Printing; Additive Manufacturing; Resource Description; Capability Description; Service Selection; Service Discovery

I. Introduction

For the efficient usage of 3D-printing resources in Cloud Manufacturing (CM) scenarios, it is necessary to schedule the existing resources. This scheduling is in accordance with the requirements of the users. 3D-printing resources are mainly 3D-printers of various types, makes and models. These 3D-printers are characterised by differing capabilities and constraints for their usage. In a cloud printing environment where these resources are considered part of a service, it is possible to compose them into new services to achieve tasks such the efficient execution of 3D-printing requests. This work offers a practical service composition framework and tool for the description required to establish service compositions within a 3D-printing service in the domain of Additive Manufacturing (AM). For this work, the applicability of the proposed resource description is analysed.

As 3D-printing encapsulates a number of different technologies, ranging from thermoplastic extrusion fabrication, over photopolymerisation to other methods, it is a prerequisite to understand these technologies. One thermoplastic extrusion based method is called Fused Deposition Modeling (FDM) (also Fused Filament Fabrication or Free Form Fabrication (FFF)). Fabrication on the basis of curing of photopolymers in a vat is called Stereolithography (SLA). Laser-based fabrication methods are either Selective Laser Melting (SLM) or Direct Metal Laser Sintering (DMLS). Other methods exist to create physical objects directly from digital models, such as Laminated Object Manufacturing (LOM). Besides the understanding of these technologies and methods, it is important to be able to describe them in a comprehensive and machine-understandable way. Furthermore, it is important to express the inherent and derived capabilities and restrictions of these technologies and machines. The different technologies do not only differ in the materials they are able to process but also in the quality that is achievable. They further differ in the geometric and structural features they can reproduce, in the cost they effect, and the means they are controlled by or programmed with. For the automated usage in a distributed service scenario, with a number of different 3D-printing resources involved, the service must be able to select an appropriate device or devices for any given user submitted task.

For the hardware providers, it is beneficial if their equipment is utilized to a high degree. This is required in order to amortise their assets on time and also to be ecologically sound [1]. For the users, such an automated and swift resource allocation is pertinent. This equates to a reduced turnaround time and also the promise of higher product quality due to optimum capability and requirements matching. For service operators, the automated resource allocation is an intrinsically motivated requirement for the operation of such a service.

With this work, a solution for the description of differing capabilities, restraints and requirements of various 3D-printing resources is provided. This solution provides an extensible, flexible, comprehensive and usable description format for the use in AM scenarios. The solution combines existing approaches for the description of resource capabilities and extends these for the usage in 3D-printing.

This work is motivated by the following four use cases:

3D-printer selection: The resource description, applied to a database of commercially available 3D-printers can serve as a purchasing guide for end-users/consumers or other potential buyers of 3D-printers [2], [3]. This will especially be the case if the information is readily available as

 ISBN: 978-1-61208-528-9

a Web-service and supports pro-active user-questioning, e.g., a wizard.

Automated facility planning: In future modular factory designs, the dynamic reconfiguration of the shop floor [4] is becoming relevant. With a machine readable resource description, layouting and planning software can place the manufacturing resources at an appropriate location.

Scheduling in 3D-printing services: In this use case the resource description is the foundation for the scheduling algorithm that selects the most appropriate available 3D-printing resource for any given processing request, based on the constraints and preferences provided by the user and derived from the model data [5], [6].

Recommender systems for CAD development: Based on the resource description a software system can support Computer Aided Design (CAD) designers with information and recommendations for geometrical and topological features within models that are manufacturable with 3D-printing resources available to a company.

This work is structured as follows: Starting with related work in Sect. II, a review of existing publications is performed. In Sect. III, the approach for the resource description is described, its underlying concepts and sources as well as the implementation and evaluation. In the Sect. IV, the implementation and its results are discussed and analysed. Lastly, the Sect. V provides a summary of this work.

II. Related Work

In the work by Pryor [7], the implementation of a 3D-printing service within an academic library is described. The system consists of two low-cost hobbyist 3D-printers and a 3D scanner. Of relevance to this work is the description of the workflow for the user handling. Pryor describes the processing workflow as purely manual with the data being deployed by the users either via a web form or email. The library staff performs sanity checking, pre-processing (i.e., positioning, slicing, machine code generation) and manual scheduling of the 3D-printer resource. The text does not provide an analysis of the time required for the staff to perform these tasks.

In the article by Vichare et al. [8], the authors propose a Unified Manufacturing Resource Model (UMRM) for the resource description of machines within the manufacturing domain. Specifically, the authors aim to describe Computer Numeric Control (CNC) machines and their associated tools in a unified way to represent the capabilities of these systems in their entirety. Their work provides a method to describe a CNC machine in an abstract sense for use in software, e.g., for simulations. As part of the collaborative peer-robot control system described in the work by Yao et al. [9], an ontology for a resource description is partially described on which we build our work. This ontology distinguishes between hardware and software resources, as well as capability and status description. The authors provide an exemplary Extensible Markup Language (XML) schema definition for such a resource description on which we extend upon. The *3D Printer Description File Format Specification* (3PP) by Adobe [10] is very relevant to this work, as it describes the 3D-printer's capabilities in XML format as deemed necessary by Adobe, presumably for the application within their software. This work contains an extensive listing of possible attributes relevant to a resource description on which we base our work. The 3PP format is limited to FDM 3D-printers. The definition includes hardware and material description but only partially caters for software support. In the publication by Chen et al. [11], the authors provide another approach to the problem of model-fabrication resource mismatch by the introduction on an abstract intermediary specification format. The authors propose this reducer-tuner model to abstract design implementations for the application to a variety of 3D-printers whereas our work proposes a 3D-printer resource description that enables the matching of suitable machines to specific model files. In the work by Dong et al. [12], the problem of scheduling in AM is handled by a rule-based management of autonomous nodes, i.e., 3D-printers. This system is based on an ontology for 3D-printing of which some excerpts are presented in this work. From this example, our work is influenced and extends on missing attributes. Yadekar et al. [13] propose a taxonomy for CM systems that are closely related to AM. This taxonomy is focused on the concept of uncertainty and only briefly discusses the taxonomical components that define the manufacturing resources. The main distinction for the authors is the division into soft and hard resource groups. In the work by Mortara et al. [14], a classification scheme for direct writing technologies, i.e., AM, is proposed. The authors define the scheme for three dimensions, namely technology, application, and materials. The properties of specific materials are discussed exemplary in brief. A listing of potential properties for the varying technologies and materials is missing.

III. Materials and Methods

From existing literature and expertise, we construct an ontology that is described in the following section. This ontology is the basis for the extension of the properties proposed that are relevant to the domain of AM. In this work, we exclude concepts like business process related capabilities, and knowledge and abstract ability related mapping, i.e., it is not possible to express certain abilities of people, teams or companies, e.g., the level of knowledge for the design of objects for AM. The properties are derived from literature and 3D-printer documentations. The following requirements are expressed to guide the generation of the ontology and properties list:

RQ1 The ontology and properties list must be flexible and extensible. Flexibility means that for specific application scenarios where only subsets of properties and relations are of interest, these must be expressible within the proposed ontology or resource description. Extensibility denotes the property to be able to incorporate future, currently unforeseen, properties of technology and materials.

RQ2 The resource description must be able to reflect temporal, local and other ranges of validity and restrictions. Conditional validity is to be reflected. With this requirement we reflect the necessity that certain properties, e.g., material strength, are only valid and guaranteed for a certain period.

RQ3 The resource description must be able to distinguish between general concepts of things, e.g., 3D-printers and materials, that form a class and its individual instantiation that might have differing properties and attributes.

In this work, the following separation of information description is performed for the resource description:

Materials: Encompasses all physical materials that are processed, or used during the digital fabrication. Also includes physical materials that are required for the digital fabrication process as indirect or auxiliary material.

Software: Encompasses all software and Information technology (IT) components that are involved in the model creation phase, the object fabrication phase or that are used for the control and management of digital fabrication equipment.

Processes: Encompasses all intangible processes, data and information that is generated, consumed, transformed or influenced by in any phase of the digital fabrication process. Business processes are part of this grouping.

Technology: Encompasses all hardware and machine equipment that is used for the object fabrication, as well as pre- and post-processing.

We exclude status information and status dependent properties from our resource description and ontology.

The resource description must be able to reflect required properties and information of all currently available 3D-printing technologies, regardless of the technology classification following any schema, such as the classification by Gibson et al. [15], the classification by Williams et al. [16] or the ISO/ASTM Standard 52900:2015 [17] classification. This work identifies common attributes between technologies and enables technology specific properties. As a guideline for the creation of the ontology and the resource description itself a distinction between object classes and their actual instances is followed. Given the example of a 3D-printer, the class is formed of all 3D-printers from a certain manufacturer and are of a certain make share a number of attributes like physical volume and number of printheads. Those general attributes might be extended by attributes pertaining to a certain 3D-printer that belongs to a user and is situated at a physical location. The general attributes might also be altered for a specific 3D-printer, as it might weight more than the original 3D-printer due to added extensions or modifications, or its build envelope is smaller than the original's due to a hardware defect.

A. Sources

Properties are extracted from datasheets from the following manufacturers and models:

3D Systems, Inc.: ProJet 7000 SD & HD, ProX 950, sPro 140, ProX DMP 200, ProX 800, ProX SLS 500, ProJet CJP 360, ProJet 1200, CubePro
Arcam AB: Arcam Q10 Plus, Arcam Q20 Plus, Arcam A2X
B9Creations LLC: B9Creator V1.2
CEL: CELRobox
Deltaprintr: Delta Go
EnvisionTEC GmbH: 3D-Bioplotter Starter Series, SLCOM1
EOS GmbH: EOS M 100, EOS M 290, FORMIGA P 110, EOS P 396, EOSINT P 800
ExOne GmbH: S-Max, S-Print, M-Flex Prototype 3D Printer
FlashForge Corp.: Creator Pro 3D
Formlabs Inc.: Form 2
LulzBot/Aleph Objects, Inc.: TAZ 6
Makerbot Industries, LLC: Replicator+, Replicator Z18
Mcor Technologies Ltd.: ARKe, IRIS HD
Optomec Inc.: LENS 450, Aerosol Jet 200
Renishaw plc.: RenAM 500M
RepRap: Prusa i3
SeeMeCNC: ROSTOCK MAX V3
SLM Solutions Group AG: SLM 125, SLM 280 2.0
Stratasys Ltd.: uPrint SE, Objet24, Dimension Elite, Fortus 380mc, Objet1000 Plus
Ultimaker B.V.: Ultimaker 3, Ultimaker 2+
UP3D/Beijing Tiertime Technology Co., Ltd.: UPBOX+
voxeljet AG: VX 200, VX 2000
WASP c/o CSP s.r.l.: DeltaWASP 20 40 Turbo

Furthermore, properties and capability attributes are extracted from publicly available slicing software (e.g., *Slic3r* [18], *Cura* [19], and *Netfabb* [20]) and acquired through experimentation. On the ontological concept itself, we refer to the work by Gruber [21] and the book by Fensel [22]. Following the distinction of ontologies by Ameri and Dutta [23], we classify our ontology as lightweight. For the construction of the ontology a list of key terms is compiled from existing glossaries and literature.

B. Properties

The following properties are identified from literature and technology documentation. These properties are listed in the appendix in order to avoid a disruption of the text flow. The provided listing is sufficient to describe relevant properties of AM machinery, i.e., 3D-printers, and the associated materials.

The properties can be further classified as either static, e.g., the serial number of a 3D-printer or its coordinate system, or dynamic, e.g., the owner or location of a 3D-printer. Dynamic properties are often dependent properties, which is a further classification applied to the properties. Dependent properties are influenced and depend upon a 3D-printer component, e.g., the nozzle and its diameter, the material, e.g., surface roughness achievable differs for materials processable or parameters selected during the 3D-printing process. This classification is not provided with this work due to brevity. The properties in the listing are for the hardware resources, i.e., the 3D-printer as well as its components and the material associated with the 3D-printer.

C. Implementation

In this section the implementation of both the ontology and the relevant core classes are described. Furthermore, information on a possible scheduling metric based on a cost estimation method and the resulting information flow in the implemented service is described.

The ontology is constructed using the protégé software version 5.1.0, see http://protege.stanford.edu/. The ontology is generated based on the properties brought forward in Sect. III-B. The guiding principle for the ontology is the flexibility of the properties that are applicable to 3D-printers, material and inherent constraints. The ontology is created based on the identified properties and derived concepts from literature and documentation.

The implementation in software to manage the specific properties of the resource description and to evaluate the applicability of the description is performed in the proposed 3D-printing cloud service by the authors [24], [25].

The implementation in the service is performed to enable provisional scheduling for 3D-printing resources based on availability, build volume and processable material type. In scheduling, some form of ordering metric must be provided. In this work, this metric is based on a proposed cost metric as described further in the text.

The cost metric is defined in [26] and serves as a prototypical implementation of cost estimation within AM.

The cost is calculated as (see Eq. 1) follows:

$$\begin{aligned} \text{Cost} = {} & (\text{Discount}(T, P, U) + \text{Profit}(U)) \\ & \times (\text{Machine} + \text{Material}(O, P, S, SO) \times \text{Factor B} \\ & + \text{Duration}(O, S, SO) \times \text{Factor U} + \text{Factor A} \\ & + \text{Factor C}(O, P)) \end{aligned} \tag{1}$$

With the following abbreviations used in the equation: 1) T for team 2) P for 3D-printer 3) U for user 4) O for object 5) S for slicer and, 6) SO for slicing options The cost for a 3D-print is dependent upon the 3D-printer selected (base cost), the material that is consumed and the time required for 3D-printing. Within the service, these attributes are user selectable for each materialtype and 3D-printer that is under the control of the user.

Based on the cost metric, scheduling is implemented in the service as described below.

In Figure 1, the processing flow for the registration of a hardware resource with the 3D-printing service is depicted. In this figure, the user dispatches a 3D-printing requirement (Job) with the service for which a number of implicit and explicit requirements and restrictions are also deposited. A hardware resource registers its capabilities with the service, that is then stored with the resource registry. The service queries the resource registry for a suitable hardware resource for a job and issues the appropriate commands for a 3D-printing execution on this resource. On completion or failure, the user issuing the job is notified.

1) Core Classes: The core classes in the ontology are described in this section. A visual representation of the ontology is depicted in Figure 2. This graph is created using the *WebVOWL* service [27].

MaterialGroup and **Material**, these classes denote the materials that are relevant for the description of the capabilities of the 3D-printing resource. The materials have an influence on a number of quality properties, e.g., the surface roughness. The materials a 3D-printing resource can process are relevant for the selection of the appropriate 3D-printing resource.

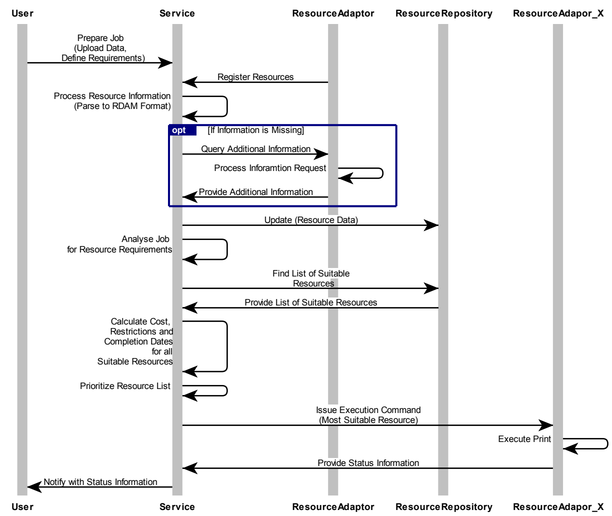

Fig. 1: Processing Flow for the Registration and Selection of a Hardware Resource

PrintingTechnology, **PrinterType**, and **Printer**, are classes to represent the underlying technology of a 3D-printing resource, e.g., a FDM based technology or a Electron Beam Melting (EBM) technology as well as the 3D-printer class which can be understood for example as a specific model line from a hardware manufacturer (e.g., the Replicator Series from Makerbot Industries). Hardware resources of a PrinterType have a number of common attributes that extend the PrintingTechnology. The Printer denotes the make of a specific PrinterType, e.g., the *MakerBot Replicator 2X* from Makerbot Industries. Instances of this Printer class have further common attributes extending the attributes of the PrinterType. Instances of the Printer class are actual 3D-printers that have further attributes like owner and a physical location.

PrinterComponent, is the class for the physical and immaterial components that are part of the specific 3D-printer. Every component can have a unbounded number of properties as described below. For example the printhead and its nozzles are components of a 3D-printer in the case of FDM technology and an electron source is a component of a EBM type 3D-printer.

Software, denotes all software that is used in the 3D-Printing Process (3D-PP). Software is used to control the 3D-printing resource, to convert files from one format into another, to prepare and process the files required for the control of the 3D-printer and to evaluate and monitor the 3D-print itself.

MProperty, this class is the generalisation of properties that

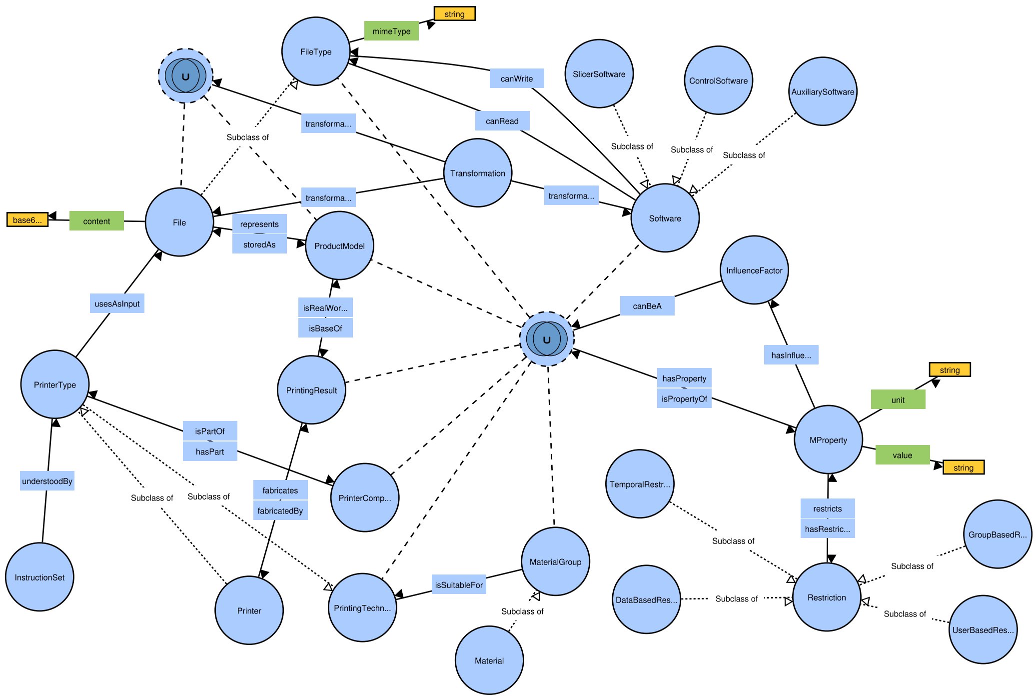

Fig. 2: 3D-printing Ontology

are applicable to either the Material, Materialgroup, PrintingTechnology, PrinterType, Printer, PrinterComponent, Software, ProductModel or File. The guiding principle for the creation of this ontology is to enable flexibility and expandability, so this generalised property can hold all properties listed above (see Sect. III-B) and future properties.

Restriction, is a class that reflects the ability to enable restrictions on MProperties as the properties can be applicable only for a specified period of time or for a certain group of people. For example the property of filament quality might be linked to a certain expiration date.

InfluenceFactor, is a class that reflects the multi-dimensional influences on properties by a defined number of factors. For example the nozzle diameter can influence the extrusion rate in case of a FDM 3D-printer.

D. Resource Description Schema

From the ontological concept, an XML schema definition is constructed which follows the principle of flexibility by encapsulation of properties in a flexible element. The property element is applicable to all relevant types of the schema, namely the PrintingTechnology, PrinterType, Printer, Printercomponent, Materialtype, and Material.

All properties are extended to allow for restrictions based on user, group or temporal conditions. The properties can be influenced by any other class of the schema to reflect interdependent relations between components. The following example justifies this construction: In the 3D-printer, the property of the material deposition rate is dependent upon the technology in use, the material processed and, in case of the FDM technology, the nozzle diameter of the extruder installed in the 3D-printer. See the following excerpt from the schema definition on the components properties and the implementation on the influencing factors:

```
<xs:complexType name="influence">
<xs:sequence minOccurs="1" maxOccurs="1">
<xs:element name="id" type="xs:ID"
  minOccurs="1" maxOccurs="1" />

<xs:choice>
<xs:element ref="tdp:MaterialType" />
<xs:element ref="tdp:Material" />
<xs:element ref="tdp:PrinterType" />
<xs:element ref="tdp:Printer" />
<xs:element ref="tdp:PrinterComponent" />
<xs:element ref="tdp:PrintingTechnology" />
</xs:choice>

<xs:element name="influenceMethod"
  type="xs:string" />
</xs:sequence>
</xs:complexType>

<xs:complexType name="validity">
<xs:sequence>
<xs:element name="id" type="xs:ID"
  minOccurs="1" maxOccurs="1" />
<xs:element name="validityCondition"
  type="xs:string"
  minOccurs="1" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>
```

 ISBN: 978-1-61208-528-9

```
<xs:complexType name="mproperty">
<xs:sequence>
<xs:element name="unit"
  type="xs:normalizedString"
  minOccurs="1" maxOccurs="1"/>
<xs:element name="description"
  type="xs:normalizedString"
  minOccurs="1" maxOccurs="1"/>
<xs:element name="value"
  type="xs:normalizedString"
  minOccurs="1" maxOccurs="1"/>
<xs:element name="name"
  type="xs:normalizedString"
  minOccurs="1" maxOccurs="1" />
<xs:element name="added"
  type="xs:dateTime"
  minOccurs="1" maxOccurs="1" />
<xs:element ref="tdp:influence"
  maxOccurs="unbounded" />
<xs:element ref="tdp:validity"
  maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>
```

IV. Discussion

The proposed resource description offers the ability to the user to select the appropriate 3D-printing resource in a scenario where restrictions for the suitable 3D-printing resources can be derived, from either the users input or from the provided data files. Within a 3D-printing service, the user is enabled to state preferences and restrictions, such as the desired quality of the 3D-printed object or cost restrictions, based on which the service itself can query appropriate hardware resources for their availability and suggest them to the user. Furthermore, based on the provided models the service can exclude certain hardware resources if they are not fitting for the task to be executed. For example, if the model file is analysed and found to contain features under a certain threshold, the hardware that is not capable of manufacturing features of this dimension are to be excluded.

A perceived problem with the flexibility of the ontology and resource description is the requirement for contextual property checking within the service itself. As opposed to strict formalities possible with the XML Schema Definition (XSD) definition, this flexibility hinders such formality checking. The 3D-printing service must be equipped with a component that is capable of evaluating the provided properties and check them for completeness, applicability and correctness. The resource description also allows for the encapsulation of third-party 3D-printing services within the 3D-printing service itself, where the capabilities of these services are regarded as a resource and described as such.

V. Conclusion

This work provides an ontology of the AM domain with extensible and flexible constructs. The derived XSD provides flexibility for extensions, based on future developments of 3D-printing hardware. The flexibility also allows for user-centric extensions and use-cases. The use case for this work is the deployment in a 3D-printing service but other use cases are also provided, such as the use within a recommender system for the design and modelling phase, or purchase recommendation systems.

In future work, it is recommended to extend the ontology to include concepts that enable the expression of immaterial capabilities and abilities, such as the expertise in certain domains, e.g., Aerospace engineering, medical engineering or bioprinting, in AM. Furthermore, it is recommended to enable the expression of proficiency in areas related to the 3D-printing lifecycle or process itself, e.g., proficiency with the design process, with the software / IT components or with legal and business concepts for AM.

Acknowledgment

The authors would like to thank Julia Holzschuh for her contributions to this work.

Appendix
Listing of Relevant Properties / Attributes

1) Operating Temperature Min/Max 2) Operating Humidity Min/Max 3) Machine Weight/Length/Height/Depth 4) Install Size Weight/Length/Height/Depth 5) Build Envelope Height/Width/Depth/Radius 6) Machine Data Connection 7) Electrical Input Rating 8) Minimum Possible Hole Diameter 9) Positioning Accuracy X/Y/Z 10) Repeatability X/Y/Z 11) Print Accuracy X/Y/Z 12) Number of Extruders 13) Nozzle Diameter 14) Temperature Extruder Min/-Max 15) Layer Thickness Min/Max 16) Movement Speed Min/Max 17) Extrusion (Movement) Speed Min 18) Print Head Acceleration Max 19) Print Bed Speed X/Y/Z Min/-Max 20) Print Bed Acceleration X/Y/Z Min/Max 21) Print Bed Temperature Min/Max 22) Binder Material 23) Processable Material 24) Processable Material Grain Size Min/Max 25) Max Object Weight 26) Lead Time Influencing Factors 27) Lead Time Formula 28) Requires Personal Attendance During Print 29) Requires Manual Interaction for Start/End 30) Resolution X/Y/Z Min 31) Operation Allowed for User/-Group 32) Maximum Achievable Surface Roughness 33) Systematic Shrinkage during Build 34) Atmosphere Pressure/-Connection/Content 35) Consumables 36) Compressed Air Supply 37) Atmosphere Consumed 38) Beam Focus Diameter 39) Laser Energy 40) Scanning Speed Min/Max 41) Laser Type 42) Power Supply 43) Power Consumption 44) Power Phase Requirement 45) Precision Optics 46) Legal Conformity Certificates 47) Workstation Requirement Ram Min 48) Workstation Requirement OS 49) Workstation Requirement CPU Min 50) Workstation Requirement Net 51) Resolution X/Y/Z 52) Number of Jets 53) Accepted File Formats 54) Number of Colors 55) Color Model 56) Manufacturer 57) Model 58) Serial Numbers 59) Object Bounding Box X/Y/Z Min/Max 60) Min Supported Wall Thickness 61) Min Unsupported Wall Thickness 62) Min Supported Wire 63) Min Unsupported Wire 64) Min Emboss Detail Width/Height 65) Min Engraved Detail

Width/Height 66) Min Escape Holes 67) Clearance 68) Enable Interlocking Parts 69) Maximum Angle for Unsupported Overhang 70) Available Infill Patterns 71) Active Cooling Extrudate 72) Hot/Cold Pause Ability 73) Requires Support Structure 74) Cathode Type 75) Vacuum Pressure 76) Material Supply Format/Packaging 77) Noise (Operation/Preparation/Idle) 78) Laser Wave Length 79) Material Deposition Mechanism 80) Number of Print Heads 81) Filament Diameter 82) Stepper Motors 83) Build Plate Material 84) Nozzle Heat Up Time 85) Build Plate Heat Up Time 86) Build Speed 87) Platform Leveling Mode 88) Laser Class/Certification 89) Peel Mechanism 90) Resin Fill Mechanism 91) Extruder Heater Cartridge Wattage/Voltage 92) Firmware Name/Version 93) Deposition Rate 94) Special Facility Requirements 95) Network Connectivity 96) Automatic Material Recognition 97) Internal Lighting 98) Enclosed Build Envelope 99) 3rd Party Material Compatible 100) Nozzle Offset X/Y/Z 101) Coordinate System 102) Printer Geometry 103) Coordinate System Origin 104) Absolute/Relative Density 105) Cytotoxicity (ISO 10993-5) 106) Melting Point 107) Magnetic Permeability 108) Electrical Resistivity 109) Specific Heat Capacity 110) Coefficient of Thermal Expansion 111) α/β Transus Temperature 112) Micro Vickers Hardness 113) Macro Rockwell C Hardness 114) Thermal Conductivity 115) Flexural Modulus/Strength 116) Tensile Modulus/Strength 117) Elongation at Break 118) Impact Strength 119) Heat Deflection Temp 120) Viscosity 121) Shore Hardness 122) Dielectric Constant/Strength 123) Volume Resistivity 124) Flammability 125) Young's Modulus 126) Yield Strength and 127) Ultimate Tensile Strength.

REFERENCES

[1] V. A. Balogun, N. Kirkwood, and P. T. Mativenga, "Energy consumption and carbon footprint analysis of Fused Deposition Modelling: A case study of RP Stratasys Dimension SST FDM," *International Journal of Scientific & Engineering Research*, vol. 6, no. 8, pp. 442–447, August 2015. [Online]. Available: http://www.ijser.org/research-paper-publishing-august-2015_page3.aspx

[2] D. A. Roberson, D. Espalin, and R. B. Wicker, "3d printer selection: A decision-making evaluation and ranking model," *Virtual and Physical Prototyping*, vol. 8, no. 3, pp. 201–212, 2013. [Online]. Available: http://dx.doi.org/10.1080/17452759.2013.830939

[3] M. Fumo and R. Noorani, "Development of an Expert System for the Selection of Rapid Prototyping and 3D Printing Systems," in *Proceedings of the 6th International Conference on Computer Science Education: Innovation & Technology*, 2015, pp. 14–18. [Online]. Available: http://dx.doi.org/10.5176/2251-2195_CSEIT15.8

[4] O. Morariu, C. Morariu, and T. Borangiu, "Shop-floor resource virtualization layer with private cloud support," *Journal of Intelligent Manufacturing*, vol. 27, no. 2, pp. 447–462, 2016. [Online]. Available: http://dx.doi.org/10.1007/s10845-014-0878-7

[5] M. Gen and L. Lin, "Multiobjective evolutionary algorithm for manufacturing scheduling problems: state-of-the-art survey," *Journal of Intelligent Manufacturing*, vol. 25, no. 5, pp. 849–866, 2014. [Online]. Available: http://dx.doi.org/10.1007/s10845-013-0804-4

[6] M. Dios and J. M. Framinan, "A review and classification of computer-based manufacturing scheduling tools," *Computers & Industrial Engineering*, vol. 99, pp. 229–249, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.cie.2016.07.020

[7] S. Pryor, "Implementing a 3d printing service in an academic library," *Journal of Library Administration*, vol. 54, no. 1, pp. 1–10, 2014. [Online]. Available: http://dx.doi.org/10.1080/01930826.2014.893110

[8] P. Vichare, A. Nassehi, S. Kumar, and S. T. Newman, "A Unified Manufacturing Resource Model for representing CNC machining systems," *Robotics and Computer-Integrated Manufacturing*, vol. 25, no. 6, pp. 999–1007, 2009, 18th International Conference on Flexible Automation and Intelligent Manufacturing. [Online]. Available: http://dx.doi.org/10.1016/j.rcim.2009.04.014

[9] Y. Yao, D. Chen, L. Wang, and X. Yang, "Additive Manufacturing Cloud via Peer-Robot Collaboration," *International Journal of Advanced Robotic Systems*, vol. 13, no. 3, pp. 1–12, 2016. [Online]. Available: http://dx.doi.org/10.5772/63938

[10] A. S. Incorporated, "3D Printer Description File Format Specification," 2014, version 1.0 draft 3. [Online]. Available: http://static.photoshop.com/content/downloads/Adobe\%203D\%20Printer\%20Profiles.pdf

[11] D. Chen, D. I. W. Levin, P. Didyk, P. Sitthi-Amorn, and W. Matusik, "Spec2Fab: A Reducer-tuner Model for Translating Specifications to 3D Prints," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 1–10, July 2013. [Online]. Available: http://dx.doi.org/10.1145/2461912.2461994

[12] C. Dong, Y. Yuan, and W. Lei, "Additive manufacturing cloud based on multi agent systems and rule inference," in *2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference*, May 2016, pp. 45–50. [Online]. Available: http://10.1109/ITNEC.2016.7560316

[13] Y. Yadekar, E. Shehab, and J. Mehnen, "Taxonomy and uncertainties of cloud manufacturing," *International Journal of Agile Systems and Management*, vol. 9, no. 1, pp. 48–66, 2016. [Online]. Available: http://dx.doi.org/10.1504/IJASM.2016.076577

[14] L. Mortara, J. Hughes, P. S. Ramsundar, F. Livesey, and D. R. Probert, "Proposed classification scheme for direct writing technologies," *Rapid Prototyping Journal*, vol. 15, no. 4, pp. 299–309, 2009. [Online]. Available: http://dx.doi.org/10.1108/13552540910979811

[15] I. Gibson, D. Rosen, and B. Stucker, *Additive Manufacturing Technologies - 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing*, 2nd ed. Springer New York, 2015. [Online]. Available: http://dx.doi.org/10.1007/978-1-4939-2113-3

[16] C. B. Williams, F. Mistree, and D. W. Rosen, "A Functional Classification Framework for the Conceptual Design of Additive Manufacturing Technologies," *Journal of Mechanical Design*, vol. 133, no. 12, pp. 1–11, December 2011. [Online]. Available: http://dx.doi.org/10.1115/1.4005231

[17] *ISO/ASTM 52900:2015 Additive manufacturing — General principles — Terminology*, ISO Std., 2016. [Online]. Available: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=69669

[18] A. Ranellucci, H. B. Andersen, N. Dandrimont, M. Hindness, P. Ledvina, Y. Sapir, M. Sheldrake, and G. Hodgson, "Slic3r – g-code generator for 3d printers," http://slic3r.org, 2011, accessed: 2016-11-28.

[19] U. B.V., "Cura 3d printing slicing software," https://ultimaker.com/en/products/cura-software, 2015, accessed: 2016-10-21.

[20] I. Autodesk, "Why netfabb?" https://www.netfabb.com, 2011, accessed: 2016-10-20.

[21] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing?" *International Journal of Human-Computer Studies*, vol. 43, no. 5, pp. 907–928, 1995. [Online]. Available: http://dx.doi.org/10.1006/ijhc.1995.1081

[22] D. Fensel, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, 2nd ed. Springer-Verlag Berlin Heidelberg, 2004. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-09083-1

[23] F. Ameri and D. Dutta, "An Upper Ontology for Manufacturing Service Description," in *ASME 2006 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 3, September 2016, pp. 651–661. [Online]. Available: http://dx.doi.org/10.1115/DETC2006-99600

[24] F. Baumann, O. Kopp, and D. Roller, "Universal API for 3D Printers," in *INFORMATIK 2016. Lecture Notes in Informatics (LNI)*, H. C. Mayr and M. Pinzger, Eds., vol. P-259. Gesellschaft für Informatik, 2016, pp. 1611–1622. [Online]. Available: http://subs.emis.de/LNI/Proceedings/Proceedings259/article155.html

[25] F. Baumann, J. Eichhoff, and D. Roller, *Collaborative Cloud Printing Service*. Springer International Publishing, 2016, pp. 77–85. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46771-9_10

[26] F. W. Baumann, O. Kopp, and D. Roller, "Abstract api for 3d printing hardware and software resources," *International Journal of Advanced Manufacturing Technology*, 2016, submitted - Under Review.

[27] S. Lohmann and S. Negru, "Vowl: Visual notation for owl ontologies," http://vowl.visualdataweb.org, 2016, accessed: 2016-10-20.

All URLs are last checked on February 9, 2017.

Why We Need Advanced Analyses of Service Compositions

Thomas M. Prinz and Wolfram Amme
Course Evaluation Service and Chair of Software Technology
Friedrich Schiller University Jena
Jena, Germany
e-mail: {Thomas.Prinz, Wolfram.Amme}@uni-jena.de

***Abstract*—The programming of classic software systems is well-supported by integrated development environments (IDEs). They are able to give immediate information about syntax and some logic failures. Although service compositions are widely used within modern systems, such a support for building service compositions is expandable. In this paper, we plead for the building of an IDE for service compositions, which enables immediate failure feedback during the development. For this, there is the need for new research activities on occurring failures and how they can be found. Since most current failure finding techniques are based on accurate approaches, e.g., state space exploration, we show in a case study that the application of accurate techniques is not a suitable solution for IDEs. In most cases, they are either too time consuming or their accurate output does not lead easily to the root of a failure. As a result, we also plead for new advanced analyses of service compositions.**

Keywords–Service Composition; Analysis; Case Study.

I. INTRODUCTION

The function of the *Course Evaluation Service* at the Friedrich Schiller University Jena is the evaluation of lectures as well as of complete courses. Especially for the evaluation of the latter, the department has to handle complex questionnaires with high adaptivity. For this case, there is no standard software, which is able to define, handle, and evaluate such questionnaires.

As part of the service, we develop a software solution — *coast* [1] — which allows handling of more complex surveys than other survey tools do. This solution is based on a service-oriented architecture and uses service compositions to define processes within the system. Figure 1 shows such an abstract service composition, which handles the logic during the execution of a survey.

As the research on service-oriented architectures has passed its 20th anniversary, we expected a wide tool and development support for service compositions. However, it is hard to nearly impossible to find lightweight tools that allow the modeling and execution of service compositions and give immediate development support. Especially, the development support is improvable regarding the programming support of modern integrated development environments (IDEs).

We can find some approaches to verify service compositions in form of business processes in the literature. A large number of those approaches concentrate on the verification of the *soundness* property [2], whereas a sound service composition cannot run into deadlocks or in undesired double executions of services. Since the soundness property is defined on the runtime behaviour of the composition, most algorithms search for undesired behaviour in a simulation. In other words, they regard the *state space* of the composition, whereas the state space defines all possible reachable states.

State space-based algorithms perform *accurate* analyses of service compositions since each found fault can appear at runtime, actually. However, as each fault is a malformed reachable state, which is caused by an *error* within the service composition, finding exactly that error given a specific fault is a hard task. We will demonstrate this in a case study on soundness and derive why it is better to use inaccurate analysis techniques similar to those used by compilers. For this, we repeat a formal and language independent model for service compositions at first — the *workflow graphs* (see Section II). Subsequently, in Section III, we show within the case study on soundness that accurate analysis approaches are not suitable to give profitable tool support. Based on this case study, we explain why it is so important to have an instantaneous development support during the modelling of service compositions (see Section IV). Eventually, this paper closes with a summary and possible future work in Section V.

II. PRELIMINARIES

In the context of graphs, we use the notions $\triangleright n$ and $n\triangleleft$ to describe the sets of incoming and outgoing edges of a graph node n, respectively.

In the rest of this paper, special directed graphs, the *workflow graphs*, are used to describe service compositions in a language independent way. Workflow graphs have different kinds of nodes for parallelisms, decisions and tasks. They were originally introduced by Sadiq and Orlowska [3]:

Definition 2.1 (Workflow Graph): A workflow graph *is a quadruple* $WFG = (N, E, s, e)$ *where* (N, E) *is a* directed graph *with a set* N *of nodes and a set* E *of edges. The graph has a single* start node s *and a single* end node e. *Furthermore, the set* N *of nodes is splitted into disjoint subsets*

$$N = \{s, e\} \cup N_{Task} \cup N_{Split} \cup N_{Merge} \cup N_{Fork} \cup N_{Join}$$

where all nodes within the same subset have the same semantics and appearance. We call each node within N_{Task} *a* task node. *The nodes of set* N_{Split} *are* split nodes *and nodes of the set* N_{Merge} *are* merge nodes. N_{Fork} *contains all* fork nodes *whereas* N_{Join} *contains all* join nodes.

Furthermore, the nodes have the following properties:

 ISBN: 978-1-61208-528-9

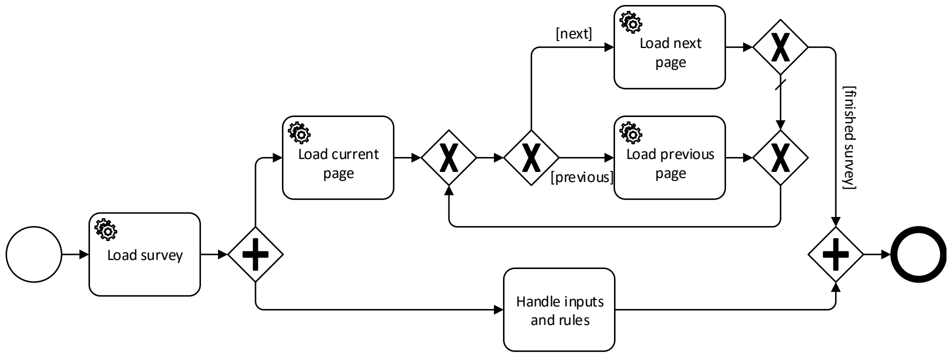

Figure 1. A service composition, which handles the logic of the execution of a survey

1) *s has no incoming and exactly one outgoing edge, whereas e has one incoming and no outgoing edge.*
2) *N_{Task}: Each node has exactly one incoming and outgoing edge.*
3) *N_{Split}/N_{Fork}: Each node has exactly one incoming and at least two outgoing edges.*
4) *N_{Merge}/N_{Join}: Each node has at least two incoming and exactly one outgoing edge.* □

For the visualization of workflow graphs, we use the same notations as the *Business Process Model and Notation* standard [4]. Therefore, tasks are illustrated as simple rectangles. Split and merge nodes are visualized by diamonds with crosses. Eventually, diamonds with pluses are used to illustrate fork and join nodes (cf. Figure 1).

These different visualizations mark the different semantics of the nodes. To describe the semantics of a node, we use a *token game* known from Petri net semantics [5]. In token games, *states* are used to describe a single execution situation. Such states can also be defined for workflow graphs [6]:

Definition 2.2 (State): A state *of a workflow graph $WFG = (N, E, s, e)$ is a* multiset *S with the basic set E. The multiset assigns a natural number t of tokens to each edge $edge \in E$ of the workflow graph, $S(edge) = t$. We say, $edge$* has *or* carries *a token in state S, if $S(edge) \geq 1$.* □

There are two important states of a workflow graph: (1) The initial state and (2) the termination state. Within the initial state, only the single outgoing edge of the start node carries a token, whereas within the termination state only the single incoming edge of the end node carries a token. In our graph visualizations, we use black dots on edges to illustrate that those edges carry a token in the current state.

In each state (except the termination state), there should be some nodes, which are *executable*, i.e., their functionality can be performed.

Definition 2.3 (Executability): Let $WFG = (N, E, s, e)$ be a workflow graph in a state S. All nodes $n \in N \setminus \{s, e\}$ are executable *in S if either (1) n is not a join node and at least one of its incoming edges has a token, or (2) each of its incoming edges carries a token. The set $\mathcal{E}xec(S)$ contains all nodes, which are executable in S.* □

If a node is executable, its execution directly follows a state transition from one state to another:

Definition 2.4 (State Transition): Assuming a state S of a workflow graph $WFG = (N, E, s, e)$ who contains an executable node $n \in \mathcal{E}xec(S)$. After the node n is executed, S changes *into the state S', written $S \xrightarrow{n} S'$. S' is defined as follows:*

$n \in (N_{Task} \cup N_{Fork} \cup N_{Join})$:
Each incoming edge in of n loses one token and each outgoing edge out of n gets a token, $S' = (S \setminus \triangleright n) \cup n\triangleleft$.

$n \in (N_{Split} \cup N_{Merge})$:
There is exactly one randomly chosen incoming edge in of n which loses a token and exactly one randomly chosen outgoing edge out of n which gets a token, $S' = (S \setminus \{in\}) \cup \{out\}$. □

Together, the executability and the state transitions form the semantics of each kind of workflow graph node. Summarized, the start and end node have no special semantics. Therefore, they are only used to mark the start and end of a workflow graph. Furthermore, each node, except a join node, is executable once there is at least one token on one of its incoming edges. A task node takes a token from its incoming edge and puts it back to its outgoing edge. Split and merge nodes perform non-deterministical choices instead: Split nodes take a token from their incoming edge and put a single token to one of their randomly chosen outgoing edges; whereas merge nodes take one token from one randomly chosen incoming edge (with a token) and put a token to their outgoing edge.

Eventually, fork and join nodes handle parallelism. Fork nodes take a token from their incoming edge and put a token on each outgoing edge. However, join nodes are only executable if each of their incoming edges has at least one token. If a join node is executed, a token is removed from each incoming edge and a single new token is placed on its outgoing edge.

As a single state can change into different states, we can define states that are *reachable* from a current state [6].

Definition 2.5 (Reachability): A state S_{to} is directly reachable *from a state S_{from} if S_{from} contains an executable node n whose execution in S_{from} leads to S_{to}.*

 ISBN: 978-1-61208-528-9

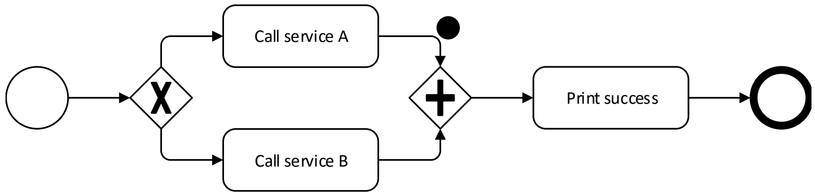

Figure 2. The success message is never printed, so there is a failure

S_{to} *is* reachable *from state* S_{from} *(we will write* $S_{from} \rightarrow^* S_{to}$*) if there is a sequence* $S_0, \ldots, S_m$*,* $m \geq 1$*, of states such that* $S_0 \rightarrow S_1 \rightarrow \ldots \rightarrow S_{m-1} \rightarrow S_m$ *and* $S_0 = S_{from}$*,* $S_m = S_{to}$*.* □

III. A Case Study on Soundness in the Context of Development Support

Before we argue for advanced analysis techniques for the immediate support during the development of service compositions, we motivate such analyses through a case study. In this case study, we consider the classic notion of *soundness*. A workflow graph is called *sound* if neither a *deadlock* nor a *lack of synchronization* is reachable from the initial state [2] [3]. A *deadlock* is a non-termination state S in which no node is executable, $\mathcal{E}xec(S) = \emptyset$. *Lacks of synchronization* are states S in which at least one $edge$ carries more than one token, $S(edge) \geq 2$.

If an execution of a workflow graph results in a deadlock or lack of synchronization, the graph's behaviour is not well defined and comprehensible. So, it is beneficial to know whether a workflow graph is sound or not. More precisely, a developer of a service composition wants to know, *why* the workflow graph runs into a deadlock or a lack of synchronization.

There are many approaches, which are able to classify whether a workflow graph is sound. The first known algorithm was introduced by van der Aalst [2]. It is based on the rank theorem [7], which can be solved in cubic time complexity regarding the size of the workflow graph [8]. However, this approach does not give any diagnostic information *where* or *why* the workflow graph is unsound [9]. For this reason, other approaches were developed, which we classify into three main approaches: (1) Model checking, (2) graph decomposition, and, finally, (3) pattern and compiler-based approaches. Examples for model checking approaches are the performed *state space explorations* by *LoLA* [10] and Woflan [11]. The *Single-Entry-Single-Exit* (SESE) approach by Vanhatalo et al. [6] is an example for a graph decomposition, whereas the anti-pattern approach of Favre et al. [12] and our compiler-based approach [13] [14] are instances for the latter class of approaches. There are many other significant approaches, however, they resemble one another in their classification.

Most techniques, like the graph decomposition and compiler-based approaches, are profitable in the context of developing service compositions although their output is inaccurate, i.e., they cannot detect faults appearing at runtime. Instead, they find incorrect structures in the compositions *may* leading to a wrong behaviour. To accentuate that inaccurate analysis techniques are suitable for IDEs, we argue in the following case study that accurate analysis techniques lead to a time expensive and hard troubleshooting.

For this case study, we consider as an example the approach of state space exploration, which dominates the literature in process verification for a long time period. Within state space exploration, the *state space* starting at the initial state is examined. Thereby, the state space is a directed graph in which each node is a state and each edge between two states S_1, S_2 means that S_2 is directly reachable from S_1. During the building of this state space, each state is checked whether it is a deadlock or a lack of synchronization. As the state space can have an exponential size depending on the size of the workflow graph, the building of the state space will be broken after the first wrong state is found. As a result, the approach indicates whether the worklow graph is sound. Furthermore, the developer gets a *failure trace*, more precisely, a path within the state space from the initial to the erroneous state.

Now, we compare the verification results of state space exploration with well known software testing terms [15]. This vocabulary makes it possible to evaluate the located faults and how they can be used for troubleshooting. Furthermore, the following comparisons motivate the usage of service composition specific advanced analyses.

A. Failures, Faults, and Errors

In software testing, there are different terms with different meanings for wrong execution states of a program. A wrong state is called a *failure* if an user of the program sees an undesired behaviour or result [15]. For example, in the workflow graph in Figure 2 we see that the last task — the printing of the success message — will not be executed since the composition runs into a deadlock in the join node. Therefore, the user is informed by the missing success message that there is a failure. Another example is a composition, which results in duplicated results as some nodes were executed twice in series caused by a lack of synchronization.

Such a lack of synchronization is the manifestation of an incorrect development of the composition. This manifestation is called a *fault* [15]. For example, the process developer may know why the user sees some duplicated results, as the developer may identify that some service calls were performed twice unnecessarily. The reason *why* the service is called twice is called an *error*. An error is the wrong human action during the development of the service

 ISBN: 978-1-61208-528-9

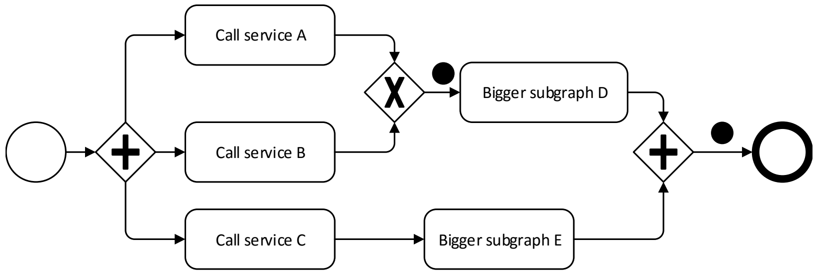

Figure 3. The distance between the fault and its error may be large

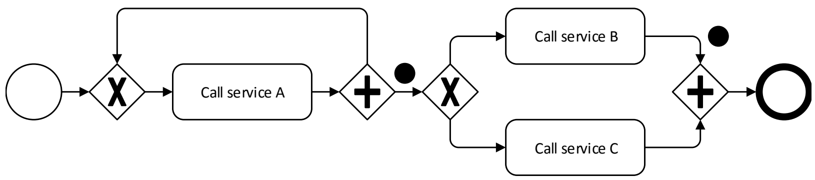

Figure 4. One fault masks another fault so that the failure may disappear

composition [15].

Obviously, to repair an erroneous service composition, a developer has to know the error instead of faults and failures. If the developer knows only the fault or the failure, it has to derive the error from the diagnostic information.

Considering the previous term definitions, each accurate analysis technique always results in a fault since it searches within the different execution possibilities of a workflow graph instead at the workflow graph itself. For this, the developer has to derive the real error, to be able to repair the composition. However, this derivation of the error is a hard task since faults can be *masked* or *disguised*. Furthermore, the *distance* between the fault and the error may be very great so that it seems impossible that a fault has its origin so early in the composition. All those different difficulties are considered in the following sub sections.

B. Fault Distance

The *distance* between a fault and its error is known as the passed time or passed program instructions until an error results in a fault [16]. The workflow graph in Figure 3 has some bigger subgraphs, which are folded as services D and E for reasons of lack of space. After the subgraph D is executed, the workflow graph will end in a deadlock state as the join on the right-hand side cannot be executed. Such a detected deadlock state is the result of an accurate soundness approach. Naturally, a developer would now search the corresponding error near the fault. Since a lot of time has passed and the workflow graph is complex caused by the subgraphs D and E, it is very hard to identify the error. A natural and simple correlation is that the difficulty of finding corresponding errors of faults grows with their distances.

C. Fault Masking

Fault masking is the situation in which one fault prevents the detection of another fault [15]. This leads to much difficulty as the faults do not necessarily cause a visible failure. Furthermore, it may happen that one fault is repaired by another one.

For example, in a program, one function should process payment information in Euro, however, it processes the data in Dollar instead. Now, another function takes the value and should translate the currency from Euro to Dollar. Coincidentally, within this function, the programmer has forgotten to implement this translation and the value is passed as-is. The result: The program has a correct behaviour as no failure happens although it does not have the desired functionality.

An example of fault masking in the context of service compositions is illustrated in Figure 4. The first part of the workflow graph (the loop) results in a lack of synchronization, whereas the second part has an obvious deadlock. However, the first part produces an endless number of tokens so that the previous lack of synchronization always prevents the latter deadlock at runtime. An accurate approach would now result in a lack of synchronization only — it is not able to detect the deadlock as it does not appear at runtime. To this end, the first fault has to be repaired *before* the deadlock appears within an accurate approach. This makes the correction of a service composition more time expensive since a necessary analysis has to run for each fault at least.

D. Fault Illusion

Fault illusion is not a classic term of software testing. We introduce it at this point because such a situation is not accurately described by the existing terms. Figure 5 exemplifies this illusion with a workflow graph. Currently, that workflow graph is within a deadlock state since there is no node, which can be executed.

 ISBN: 978-1-61208-528-9

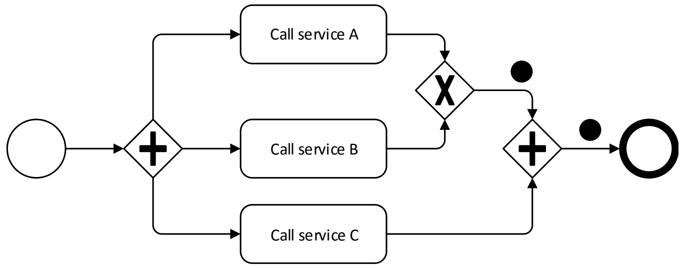

Figure 5. One fault produces another fault so that there is the illusion of an error, which does not exists

An accurate analysis technique could provide this deadlock state. However, if the developer of the service composition takes a closer look at the workflow graph, it will not find a good fitting error of the deadlock. This happens for the reason that the deadlock is caused by a lack of synchronization: The left-hand side fork node has two upper outgoing control flows that are not synchronized by a join node. Only a merge node combines both flows, which possibly results in a lack of synchronization on its outgoing edge. Nevertheless, if, e.g., service A needs much more time as service B, the control flow of service B reaches the join node on the right-hand side before the control flow, which performs service A. Because of this, the join node can be executed before the lack of synchronization appears. Then, however, the workflow graph runs into a deadlock although there is the error of a wrong control flow synchronization.

So, in short, a fault illusion is the appearance of a fault although the errors of other faults cause it. The finding of such a fault illusion is a very hard task in big service compositions. In this context, accurate analysis techniques are not suitable for error identification.

E. Fault Blocking

In software testing, *fault blocking* is the condition in which a fault blocks the further failure detection [17]. In accurate approaches, it is easy to see that it is not possible to detect faults *after* a deadlock since there is no further reachable state. As a result, it is not possible to detect all *errors* within a service composition. Hence, fault blocking makes the error detection time expensive since a necessary analysis has to run at least for each fault (which can be an arbitrary large number in the case of lacks of synchronization).

Another difficulty of fault blocking is that one fault may result in another fault. This is linked to fault illusion. In Figure 6, we see a simple workflow graph in which a split node causes (local) deadlocks in the upper and lower join nodes. However, as we can also see, the deadlock of the lower join node is caused by the deadlock of the upper one, i.e., if the upper join node would be a merge node, the deadlock of the lower join node disappears. Therefore, the deadlock of the lower join node is the result of the blocking of a control flow of the upper join node. Since an accurate fault finding approach like state space exploration may return the deadlock of the lower join node, it is hard to find its error.

IV. Development Support During the Creation of Service Compositions

In the last section, we have demonstrated in a case study that accurate analysis techniques for tool support during the development of service compositions is not suitable. Now, we argue on the base of this case study, why it is important to support the modelling of compositions.

The development of service compositions is an error-prone task just like the development of software systems. As for the latter exist IDEs, the tool support for the development of service composition is expandable. That is surprising since there is a substantial common ground between both: (1) There is data information passed through variables and (2) there is a flow graph, which represents the structure.

Besides those similarities there are some serious differences making the adaption of analyses from classical software development to service compositions difficult: (1) In most cases, service compositions are developed by the use of visual modeling languages, e.g., *BPMN* and Event-driven process chains [18]. Visually modeled compositions often result in unstructured workflow graphs, e.g., approx. 60% of all real world processes taken from IBM Zürich [19] are unstructured. Unfortunately, most known fast analysis algorithms of compiler theory work only for structured graphs.

(2) A second major difference between the development of software systems and service compositions is the ability to model explicit parallelism within service compositions. Since most algorithms for program analysis cannot be applied to parallel programs, they must be adapted [20]. Lee et al. [21] introduced the *Concurrent Static Single Assignment* (CSSA) form making it possible to use algorithms of sequential programs for parallel software. Unfortunately, the building of the CSSA form requires knowledge about possible race conditions to ensure high quality analysis results. The derivation of race conditions, however, is difficult for unstructured workflow graphs [21].

The major advantages of well-known analyses used in modern IDEs for software development are the extensive diagnostic information and the possibility to find possible failures along the whole program instead of only finding first reachable failures from the start. For example, imagine we use a variable a at line 10 and a variable b at a subsequent line 20 in a program, however, we have forgotten to define both variables before. Now, most modern IDEs will give us a feedback for *both* undefined variables although the variable

 ISBN: 978-1-61208-528-9

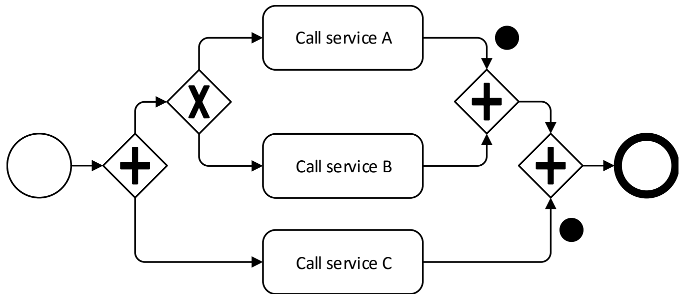

Figure 6. One fault blocks another fault

a appears always before b.

Conversely and as motivated in our case study, most analyses for service compositions do use accurate fault finding techniques, which can only find first appearing faults since afterwards the program is within a dirty state. Furthermore, the accurate finding of faults is more time consuming then performing safe over-approximations as done by classical compiler algorithms. Finally, as we have shown previously in our case study, such accurate fault finding techniques like state space exploration have some serious disadvantages during the reparation of malformed service compositions.

Summarized, we plead for an adaption of fast and well-known analysis techniques of modern IDEs to the development of service compositions. Furthermore, we argue for the development of advanced analysis techniques especially for service compositions to solve composition-specific problems. In this context, we also plead for a first real compiler for service compositions, which enables those analyses as well as the transformation of service compositions into runnable applications [22]. Such runnable applications can be executed, e.g., in a virtual machine [23].

V. Conclusion and Future Work

In this paper, we have recommended for the introduction of advanced analyses for service compositions. In this context, we have shown in a case study that accurate analysis techniques can result in an unprecise and time consuming error detection. This makes it difficult to repair a defect composition.

Our proposal is to apply well-known algorithms of compiler theory to the theory of service compositions. Therefore, some basic problems have to be solved in the future: (1) Some compiler algorithms have to be adapted to unstructured graphs. Furthermore, (2) there is the need for an algorithm to find races between two variables in unstructured workflow graphs. Then, it is possible to perform the CSSA building algorithm on service compositions, which enables other algorithms from compiler theory in a parallel context. Finally, (3) there must be a research infrastructure like a compiler to collect and apply new developed algorithms to service compositions. Such a compiler has to be connected with an IDE to support the building of compositions.

References

[1] Course Evaluation Service, "coa.st," website, visited on January 30th, 2017. [Online]. Available: http://www.coast.uni-jena.de

[2] W. M. P. van der Aalst, "A class of Petri nets for modeling and analyzing business processes," Eindhoven University of Technology, Eindhoven, Netherlands, Computing Science Reports 95/26, 1995, technical Report.

[3] W. Sadiq and M. E. Orlowska, "Analyzing Process Models Using Graph Reduction Techniques," Information Systems, vol. 25, no. 2, Apr. 2000, pp. 117–134.

[4] Object Management Group (OMG), "Business Process Model and Notation (BPMN) Version 2.0," OMG, Jan. 2011, standard. [Online]. Available: http://www.omg.org/spec/BPMN/2.0

[5] C. A. Petri, "Communication with Machines (Kommunikation mit Maschinen," Ph.D. dissertation, Faculty for Mathematics and Physics, Technische Hochschule Darmstadt, Bonn, Jul. 1962.

[6] J. Vanhatalo, H. Völzer, and F. Leymann, "Faster and More Focused Control-Flow Analysis for Business Process Models Through SESE Decomposition," in Service-Oriented Computing - ICSOC 2007, Fifth International Conference, Vienna, Austria, September 17-20, 2007, Proceedings, pp. 43–55.

[7] J. Desel and J. Esparza, Free Choice Petri Nets, ser. Cambridge Tracts in Theoretical Computer Science, C. van Rijsbergen, S. Abramsky, P. H. Aczel, J. W. de Bakker, J. A. Goguen, Y. Gurevich, and J. V. Tucker, Eds. Cambridge, Great Britain: Cambridge University Press, 1995, no. 40, iSBN 0-521-46519-2.

[8] P. Kemper and F. Bause, "An Efficient Polynomial-Time Algorithm to Decide Liveness and Boundedness of Free-Choice Nets," in Application and Theory of Petri Nets 1992, 13th International Conference, Sheffield, UK, June 22-26, 1992, Proceedings, pp. 263–278.

[9] C. Favre and H. Völzer, "Symbolic Execution of Acyclic Workflow Graphs," in Business Process Management - 8th International Conference, BPM 2010, Hoboken, NJ, USA, September 13-16, 2010. Proceedings, pp. 260–275.

[10] K. Schmidt, Application and Theory of Petri Nets 2000: 21st International Conference, ICATPN 2000 Aarhus, Denmark, June 26–30, 2000 Proceedings. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, ch. LoLA A Low Level Analyser, pp. 465–474.

[11] H. M. W. E. Verbeek, T. Basten, and W. M. P. van der Aalst, "Diagnosing Workflow Processes using Woflan," The Computer Journal, vol. 44, no. 4, Sep. 2001, pp. 246–279.

[12] C. Favre, H. Völzer, and P. Müller, "Diagnostic Information for Control-Flow Analysis of Workflow Graphs (a.k.a. Free-Choice Workflow Nets)," in Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings, pp. 463–479.

[13] T. M. Prinz and W. Amme, "Practical Compiler-Based User Support during the Development of Business Processes," in Service-Oriented Computing - ICSOC 2013 Workshops - CCSA, CSB, PASCEB, SWESE, WESOA, and PhD Symposium, Berlin, Germany, December 2-5, 2013. Revised Selected Papers, pp. 40–53.

[14] T. M. Prinz, N. Spieß, and W. Amme, “A First Step towards a Compiler for Business Processes,” in Compiler Construction - 23rd International Conference, CC 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings, pp. 238–243.

[15] I. C. Society, “IEEE Standard Glossary of Software Engineering Terminology,” IEEE Standard 610.12-1990, Dec 1990, pp. 1–84.

[16] I. Sommerville, Software Engineering, 8th ed. Munich, Germany: Pearson Studium, Apr. 2007.

[17] M. A. Friedman and J. M. Voas, Software Assessment: Reliability, Safety, Testability, 1st ed., ser. New Dimensions In Engineering Series. New York, USA: John Wiley & Sons, Inc., Aug. 1995, vol. Book 16.

[18] G. Keller, M. Nüttgens, and A.-W. Scheer, “Semantic Process Modelling Based on ”Event-driven Process Chains (EPC)” (Semantische Prozessmodellierung auf der Grundlage ”Ereignisgesteuerter Prozessketten (EPK)”),” Institut für Wirtschaftsinformatik, Saarbrücken, Germany, Veröffentlichungen des Instituts für Wirtschaftsinformatik 89, 1992, technical Report. [Online]. Available: http://www.iwi.uni-sb.de/iwi-hefte/heft089.zip

[19] IBM, “IBM Research - Zurich: Computer Science,” website, visited on January 30th, 2017. [Online]. Available: http://www.zurich.ibm.com/csc/bit/downloads.html

[20] S. P. Midkiff and D. A. Padua, “Issues in the Optimization of Parallel Programs,” in Proceedings of the 1990 International Conference on Parallel Processing, Urbana-Champaign, IL, USA, August 1990. Volume 2: Software., pp. 105–113.

[21] J. Lee, S. P. Midkiff, and D. A. Padua, “Concurrent Static Single Assignment Form and Constant Propagation for Explicitly Parallel Programs,” in Languages and Compilers for Parallel Computing, 10th International Workshop, LCPC’97, Minneapolis, Minnesota, USA, August 7-9, 1997, Proceedings, pp. 114–130.

[22] T. M. Prinz, T. S. Heinze, W. Amme, J. Kretzschmar, and C. Beckstein, “Towards a Compiler for Business Processes - A Research Agenda,” in SERVICE COMPUTATION 2015: The Seventh International Conferences on Advanced Service Computing, pp. 49–54.

[23] T. M. Prinz, “Proposals for a Virtual Machine for Business Processes,” in Proceedings of the 7th Central European Workshop on Services and their Composition, ZEUS 2015, Jena, Germany, February 19-20, 2015., pp. 10–17.