Knowledge-based Tool for Software Process Assessment and Improvement

Maria Helynne Lima Silva, Ana Carla de Carvalho Correia, Carlos Eduardo da Silva Costa, Leandro Dias da Silva, Rodrigo de Barros Paes
Universidade Federal de Alagoas
Maceió, Brazil
{mariahelynne, cc.anacarla, neo.edu.costa, leandrodds}@gmail.com, rodrigo@ic.ufal.br

Rodrigo Aiello Praes, João Manoel Silvestre de Sousa, Gustavo Robichez de Carvalho
Pontifícia Universidade Católica do Rio de Janeiro
Rio de Janeiro, Brazil
r.praes@gmail.com, joaomss@primeup.com.br, guga@les.inf.puc-rio.br

Abstract - **Although many organizations are aware of the importance of using well-defined and organized software development process, they face the problem of how to define and institutionalize it in practice. In order to solve these problems, several process models, maturity models and quality standards have been developed, but the variety of disciplines, methodologies, and best practices is large. This amount of information leads to an overload and can make the task of defining a software process complicated and expensive. To deal with scenario, this paper proposes an approach with two main goals: (i) to develop a model for organizing the knowledge on software engineering; (ii) to develop a software tool to support the model. Once the knowledge base becomes accessible through a tool, organizations can use it as a guide to a software quality improvement program.**

Keywords-Software Quality Improvement; Software Quality Assessment; Knowledge Based Tool.

I. INTRODUCTION

One of the most important factors for the quality of a software product is its development process. A well-defined process helps organizations to follow their schedules, budget and achieve the expected product quality [1]. A standardized process can reduce the room for human mistakes.

Although many organizations are aware about the importance of using well defined and organized software development process, they face the problem of how to define and institutionalize it in the organization.

In order to solve these problems, several process models, maturity models and quality standards have been developed. Typically, these models contain the knowledge acquired by a number of real software development and it is structured through a number of best practices and examples.

The variety of disciplines, methodologies, best practices, is increasing. This amount of information may lead the task of defining a software process to a complicated and expensive problem. Moreover, these models are available in an abstract and scattered way in books, websites, among others. That makes the use of this information even harder for organizations. The Guide to the Software Engineering Body of Knowledge [2] is an example of how this knowledge has been organized. The purpose of the guide is to describe what portion of the Body of Knowledge is generally accepted, to organize that portion, and to provide a topical access to it. The Guide should not be confused with the Body of Knowledge itself, which already exists in the published literature.

To deal with scenario, this paper proposes an approach with two main goals: (i) to develop a model for organizing the knowledge on software engineering, that should allow representing any reference model, such as CMMI-Dev 1.2 (Capability Maturity Model Integration for Development) [3], ISO 15288:2008 (International Organization for Standardization) [4], XP (Extreme Programming) [5] or Scrum practices [6]. (ii) To develop a tool to support the model. The tool should be able to maintain the information through insertion, removal and update, providing a knowledge base of best practices found in literature.

Once the knowledge base becomes accessible through a tool, organizations can use it as a guide to a software quality improvement program. The tool is able to diagnose the riskiest disciplines and provide a complete step-by-step quality improvement plan. The tool is also independent of the evaluation methodology, such as SCAMPI (Standard CMMI Appraisal Method for Process Improvement) [7], or maturity model used as reference, such as CMMI-Dev [3].

This paper is organized as follows. Section II describes the knowledge base model. Section III describes the tool that has been developed. Section IV presents some related work. Section V presents a case study where the tool was actually applied in a simulated scenario. Finally, Section VI briefly discusses the results obtained in the study case and presents the conclusion and future work.

II. THE KNOWLEDGE BASE MODEL

Each element of Figure 1 and the relationships between them are described below. The examples given are for the reference model CMMI-Dev.

CMMI is a process improvement maturity model for the development of products and services. It consists of best practices that address development and maintenance activities that cover the product lifecycle from conception through delivery and maintenance.

CMMI can be used to guide process improvement across a project, a division, or an entire organization. It helps integrate traditionally separate organizational functions, set process improvement goals and priorities, provide guidance for quality processes, and provide a point of reference for appraising current processes [3].

In Figure 1, <Discipline> represents a set of disciplines of Software Engineering, such as Software Requirements, Project Management Software, etc.. Thus, each discipline may be associated with one or more activities in <Activity>. For example, Software Requirements has activities such as Elicit Needs, Non-Functional Requirements, Change Control, among others.

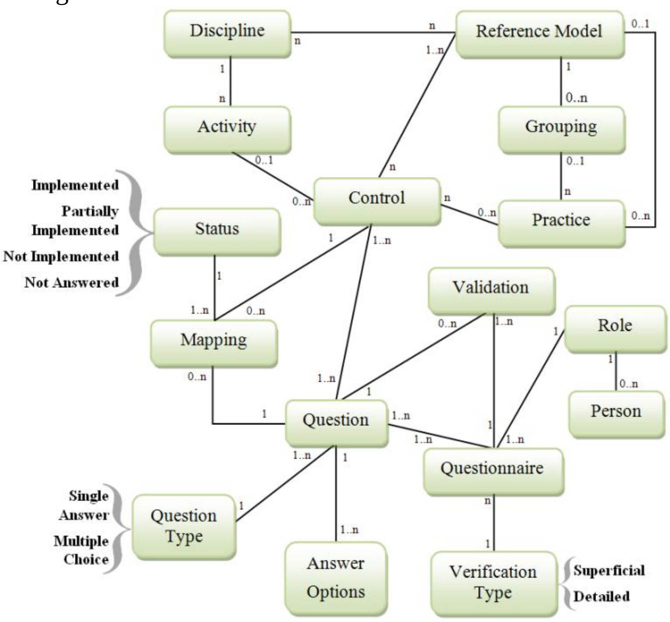

Figure 1. Knowledge base (KB) model

Moreover, <Discipline> has a many-to-many relationship with <Reference Model>, which maintains a set of reference models, for example, CMMI, XP, MPS.BR (Brazilian Software Process Improvement) [8]. Each <Reference Model> describes best practices in software engineering, so it may be related to many <Grouping> or <Practice>.

An example of <Grouping> is Requirements Management (RM). RM is concerned with managing the requirements of the project's product components and identification of inconsistencies between those requirements.

One of the RM's practices is "Requirements Changes Management" since, during the project, requirements may change for a variety of reasons. It is essential to manage these additions and changes efficiently and effectively.

A <Person> has his <Role> in a company. Examples of roles are Manager, Analyst and Developer. The roles can be used to generate specific questionnaires to each company employee.

Thus, a <Questionnaire> is a set of questions filtered according to one or more disciplines, roles and <Verification Type> that determines if the questionnaire will be "Superficial" or "Detailed".

A <Question> may be part of a series of <Questionnaire> and may have several <Answer Options>. A <Question> has a <Question Type> which means a question can be either "Single Answer" or "Multiple Choice".

The <Control> is a checkpoint. It refers to an activity, one or more reference models and one or more practices. The establishment of these relationships makes possible to verify whether the best practices are being applied within the company. The control has a P index (probability), an S index (severity) and an R index (relevance). The product of P*S*R is called PSR index. The general idea of PSR is to indicate quantitatively the risk level if the control is not implemented [9]. The values of P and S should be given by the software engineering expert during the registration of the control. The value of R is determined according to the needs of each company. Possible values and meanings of each index are shown in Table I.

TABLE I
CLASSIFICATION OF PROBABILITY, SEVERITY AND RELEVANCE VALUES

Index Value	PROBABILITY The possibility of the threat causing quality problems	SEVERITY The consequence of the quality problem	RELEVANCE The impairment in the organization	
5	Almost certain (P ⩾ 95%)	Extremely affects quality	Can affect the entire company and the losses are extremely high	Very high
4	Very likely (65% ⩽P< 95%)	Very seriously affects quality	Can affect one or more of the company's business and losses are high	High
3	Likely (35% ⩽P< 65%)	Seriously affects quality	Can affect a part of the company's business and the losses will be reasonable	Medium
2	Unlikely (5% ⩽P< 35%)	Minor affects quality	Can affect a small and specific part of the company's business and the losses will be low	Low
1	Very unlikely (P < 5%)	Hardly affects quality	Can affect a very small and specific part of the company and the losses will be negligible	Very low

Each <Question> should be associated with one or more <Control> in order to investigate the implementation of the practices, activities or disciplines of a reference model.

<Mapping> is a script question and also relates to <Control>. <Mapping> determines what alternatives of each question must be marked so that the <Status> of the <Control> is determined as "Implemented", "Partially Implemented", "Not Implemented" or "Not Answered".

In order to better understand how the controls are used, the Figure 2 describes the use of a control CMMI in tool. It accentuates the elements with descriptions corresponding the CMMI model. In the figure, the <Role> element is instantiated as "Developer". In this way, it is possible to generate a specific questionnaire for this role. The developers will answer the questionnaire and their responses will be analyzed. This questionnaire will contain the question "The clarification of their doubts about the impact that a change can cause is possible because: ". This question is used to verify the application of the control: "Requirements are managed and inconsistencies with project plans and work products are identified". The answers will indicate the level of how implemented the control is, which can assume one of the following values: "Implemented", "Partially Implemented", "Not Implemented" or "Not Answered". As the control is related to the practice "Manage Requirements Changes", it is possible to conclude if it has been applied correctly according to the chosen reference model (CMMI-Dev).

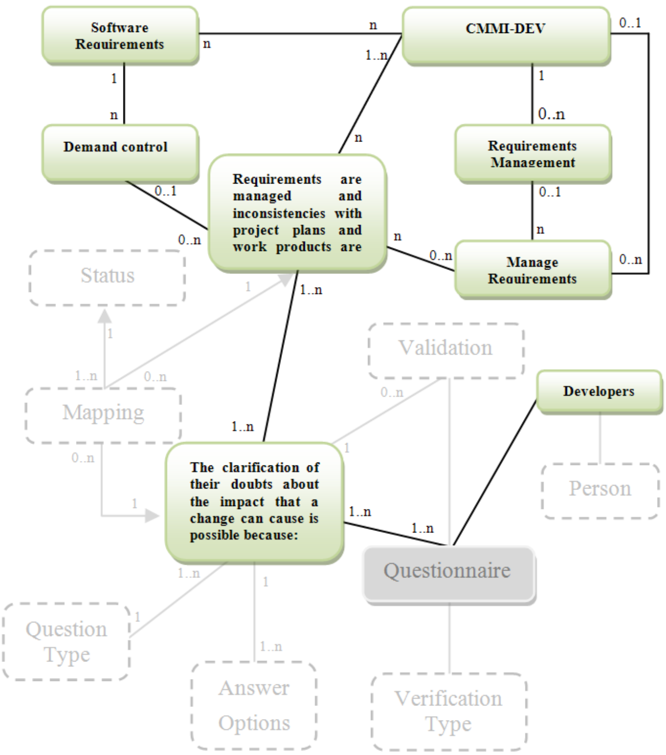

Figure 2. Example of an instantiation of the KB model using a CMMI-Dev control.

III. A SOFTWARE TOOL TO SUPPORT THE MODEL: SÁBIO

Sábio[1] is a web based tool which has been developed in this work. It has two main use cases: (i) maintain a software engineering knowledge base; and (ii) use the knowledge base to evaluate the software process of a given organization [10].

This tool aims to support the model described in Section II. The database contains the entities of the reference model, such as disciplines, practices, activities, roles, questions and controls. The tool is able to generate and send questionnaires to employees of a registered company and, based on the responses, evaluate, in a flexible way, the company's development process from the viewpoint of the employees' roles.

This section shows, step by step, the usage scenario, which is illustrated in Figure 3 and Figure 4. First, we will discuss the usage scenario of the first goal, which consists of maintaining the knowledge base (KB). After that, it will be discussed about how the evaluation process is performed, the second goal.

A. Maintain a software engineering knowledge base

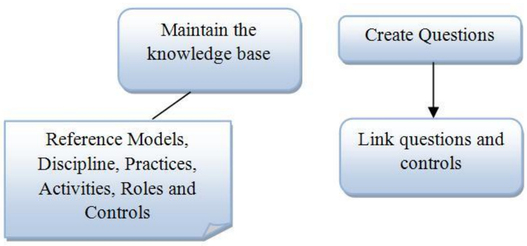

Figure 3. Usage scenario: maintaining the knowledge base.

[1] In Portuguese, the word Sábio is used to designate a wise person.

In order to achieve the first goal, software engineering experts are responsible for two main tasks. One is to build a knowledge base by ensuring quality and integrity. For this purpose, they will register reference models, practices associated with these models and disciplines of software engineering. The tool provides use cases to create, read, update and delete each of these items.

The other task of software engineering experts is to create questions in such a way that the answers extract some information about the development process of the company. When the experts create the question, they have to define its description and answer options. They must associate the question with a discipline and a role, so it is possible to create filters for the generation of the questionnaires. They must also link each question to one or more controls, and for each, write a rule, which will show the status of control.

For example, the expert registers a question and associates it with a control, based on his own knowledge he can create the following rules as scripts:

Question (example): For processes (description of proceedings, techniques, coding standards and templates, etc.) used by you:

a) [ ] There are processes and templates that describe and support the activities that I do. All documentation of these processes is available for use.

b) [ ] There is not a defined process, but we use techniques and practices that support the activities of analysis.

c) [ ] I have means to report my feedback about the activities that I do.

d) [ ] Improvements and changes are implemented in the processes, templates, techniques and practices that support the activities that I do.

e) [ ] Depending on the needs of the project, the templates, the patterns and techniques are adjusted.

f) [ ] I do not have information to answer this question

g) [ ] None of the above.

Control #1 (example) - A program to improve organizational processes should be implemented.

(f OR g) -> Not answered
(c AND d) -> Implemented
(c OR d) -> Partially Implemented
ELSE -> Not Implemented

Control #2 (example) - A useful set of organizational process assets should be established.

(f OR g OR (a AND b)) -> Not answered
(a AND e) -> Implemented
(a OR b OR e) -> Partially implemented
ELSE -> Not Implemented

In this question, two controls are being evaluated, both are related to the activity of Organizational Process Focus (OPF) and they are in accordance with the reference model CMMI-Dev. The status of the control is defined according to the chosen options. In this case, for the control #1, if an employee chooses the options 'f' or 'g', it means that this control was not answered. If the employee selected letter 'c' and 'd', it means that this control has been implemented. If 'c' or 'd' were selected, the control is partially implemented.

In control #2 case, the choices 'f' or 'g', or 'a' and 'b do not answer the question or doesn't have sense, so the control is not answered. The options 'a' and 'e' indicate that the control is implemented. If ‘a’, ‘b’ or ‘e’ were chosen, the control is partially implemented.

In the case that the chosen answers do not apply in either case, it is considered that the control is not implemented. Sometimes, it is possible to find more than one status for one control, in this case, it is considered the lowest level.

B. Evaluate the software process

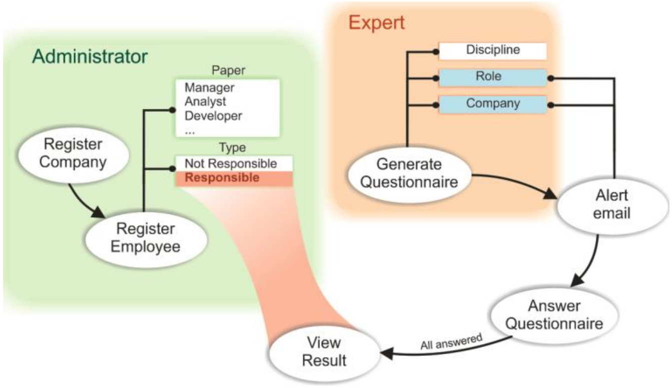

Figure 4. Usage scenario: evaluation process

For the evaluation process of a company, it is necessary to register the company in the database. This is a system administrators’ task.

After the company registration, the employees can be registered. Thus, each employee is linked to a company and to the roles that he performed within the company. The employees are the system users that will be able to answer the questionnaires that are generated by the software engineering experts. An employee may be classified as responsible for the company, this gives him the permission to consult the reports in the end of the assessment.

A questionnaire is a set of questions filtered by disciplines and roles. The software engineering experts generate the questionnaires depending on the needs of the company which has been evaluated. Notice that the questions are already saved in the knowledge base and previously linked to a discipline, roles and several controls.

In the generate questionnaire use case, the expert chooses the disciplines and the role and indicates one or more company to answer the questionnaire. First, the questions are filtered by the disciplines and then by the role. The questions bring the associated controls, for each one, it is assigned the appropriate value of the R index for the company.

All employees of the chosen companies who perform the chosen role are alerted by email that there is a new questionnaire to answer. Once each one log in to the system they will find the questionnaires and they must answer them.

After the answers, the employee clicks on the submit button and each chosen answer is saved in the database. When all the employees of a company answer the questionnaires, the responsible employees are alerted by email that the reporting is now available. Then, the responsible employee logs in to the system and can view the results in a flexible manner, for example, you can generate a report by role, by activity or by the reference model.

The assessment is based on the counting of the controls. Depending on the chosen answers, which are saved in the database, the control status can be determined, thus it is possible count the number of controls according to their status for each answered questions by every employee. In other words, it is made a counting of how many controls are implemented, how many are partially implemented, how many are not implemented and how much were not answered.

Since each control was associated with a discipline, and each questionnaire is directed at a role, it is possible to make this counting in flexibly way. For example, it is possible to analyze the status according to the managers or to have the sum of the controls which are implemented relative to the discipline of Software Requirements.

With the values of P, S and R associated with controls and with the amount of controls for each status, we can determine, using formulas, what activities have the lowest security levels, and the lowest compliance levels. These index values can be calculated in accordance with the equations (1) and (2). So, the activities with the lowest rates must be raised by the report as priorities to improve the development process.

Call:

$$Total_{IC\_PSR} = IC\_PSR + 0{,}5\ PIC\_PSR$$
$$Total_1 = EC\_PSR - NAC\_PSR$$

$$Security\ index = \begin{cases} 0, & if\ Total_1 = 0 \\ \dfrac{Total_{IC\_PSR}}{Total_1}, & if\ Total_1 \neq 0 \end{cases} \quad (1)$$

where:

IC_PSR means the sum of implemented controls’ PSR
PIC_PSR means the sum of partially implemented controls’ PSR
EC_PSR means the sum of evaluated controls’ PSR
NAC_PSR means the sum of not answered controls’ PSR

Call:

$$Total_2 = EC - NAC$$

$$Compliance\ index = \begin{cases} 0, & if\ Total_2 = 0 \\ \dfrac{IC}{Total_2}, & if\ Total_2 \neq 0 \end{cases} \quad (2)$$

where:

EC means the amount of evaluated controls
IC means the amount of implemented controls
NAC means the amount of not answered controls

IV. RELATED WORK

There are some commercial tools with similar goals to the Sábio, such as CMM-Quest [11], Appraisal Wizard [12], and IME Toolkit (Interim Maturity Evaluation Toolkit) [13]. There are also academic tools such as Evaluación Asistida de CMMI-SW (Assisted Evaluation of Capability Maturity Model Integration-Software Engineering) [14] and SPQA.web [15].

CMM-Quest, produced by HM & S IT-Consulting, is a self-appraisal tool for software development organizations to evaluate and analyze their software development processes. The main objective is to support informal assessments based methods (class B and C[2]). The method does not support

[2] The SEI has three classes of methods of evaluation:
- Class A is the most complete, most accurate results, providing a greater understanding of strengths and weaknesses of the organization. The only example of this class is the SCAMPI method.

SCAMPI (Class A). For evaluations, the model supports CMMI-SE/SW/IPPD/SS (Capability Maturity Model Integration - Systems Engineering / Software Engineering / Integrated Product and Process Development / Supplier Sourcing) [16].

Appraisal Wizard, developed by the Integrated System Diagnostics Incorporated is another tool designed to help a team of developers in the assessment of an organization. Considering the work planning, data collection, evaluation team, and generating results for the organization and evaluating these results. It supports practically all models published by SEI (Software Engineering Institute), including CMMI-SE/SW/IPPD/SS model in both representations. It also supports multiple methods of assessment including SCAMPI.

The tools CMM-Quest and Appraisal Wizard work as a repository for collecting information through an assessment. Each piece of information (evidence or opinions of strengths and weaknesses found) are classified and associated with one or more quality standards. Throughout the evaluation process, these tools are used to store data and identify practices that have been implemented.

IME toolkit allows assessments according to the model CMMI-SE/SW. The evaluations include assigning numerical values to the practices. Based on this, the tool generates scores for the process areas. It does not provide support for the SCAMPI assessment method or a detailed evaluation because it is not a tool itself, but a set of Excel spreadsheets.

In [14], the authors propose a tool that provides support for SCAMPI based evaluation. It is possible to register the practice of the CMMI model. The practices are grouped by process areas. The tool is also able to provide compliance reports with both CMMI level 2 and 3.

SPQA.web allows the evaluation of a software development process of an organization. The tool supports the assessment of some process areas of CMMI model and standard ISO/IEC 12207:2002 [17].

Although the related approaches in this section share common goals with this work, there are still some limitations that need to be addressed: (i) Reference model: one of the goals of the proposed approach is to design and build a software engineering knowledge base that is not tied to a particular reference model, as opposite to the most of the related approaches. Then, we provide a higher level model to represent practices that is independent from a particular reference model. The relationship between a generic practice and a reference model is established after the registration of the practice in the base. Furthermore, each practice may be related to more than one reference model. (ii) In the presented related work, there is a consultant that performs the diagnosis. The consultant uses the tools only to register the results of the diagnosis. The proposed work tries to systematize the generation of such diagnosis. In the current version, the way Sábio performs the diagnosis is by using questionnaires and collecting the answers from the stakeholders in order to generate the reports. (iii) The level of details provided in the diagnosis should be configurable. It may be the case that an organization wants to perform a quick and shallow diagnosis. There is also the case that an organization wants to perform a deep and detailed diagnosis. Therefore, the tools must have ways to register in the base both a detailed or superficial practice. Sábio deals with this problem by providing two different levels of practices and consequently two different levels of questionnaires. (iv) When an appraisal is performed, different stakeholders, playing different roles in the development process are involved. Then, the questions that should be asked to each different role should also be different. When a user registers a question in Sábio, he should also select the role that the question should be asked to. (v) Most of the related work provides two outputs: compliant or noncompliant. However, it would be useful to some organizations if the tool provides a report that contains practices that are being followed and the practices that should be followed. Furthermore, even for the same practice, it may have a higher importance in an organizational context than other. For example, an air traffic control company would give a higher relevance to practices related to tests and specification than other company that produces payroll software.

- Class B: A method on a smaller scale, also called mini-appraisal. It goes into less detail than in the class A, and requires less effort.
- Class C: is the least intensive of the three, also called micro-appraisal. Gives some simple idea for the practices employed in an organization.

The only method endorsed formally by the SEI is SCAMPI (Class A). Methods B and C do not have a formal specification by the SEI, leaving its implementation by the concerned.

V. CASE STUDY

To demonstrate the applicability of the tool that has been presented, it was proposed a case study in which three employees of a company were submitted to use the tool. They answered questionnaires for the assessment of the Software Requirements discipline in the development process within the company.

The team was composed of two analysts and one manager, which will be referenced by the names Analyst#1, Analyst#2 and Manager#1.

The case study was divided into four steps:

Step 1 (Company and Employees Registration): In this step, the system administrator registers the company and the employees that will answer the questionnaires.

Step 2 (Questionnaire Preparation): At this stage, the tool was used for generating questionnaires. In this case study, a questionnaire was created to evaluate only the activities of the discipline of Software Requirements.

Step 3 (Assessment and Information Collection): In this step, the team members assessed answered the questionnaire.

Step 4 (Reports Generation): The reports are generated based on the chosen answers and on the information of each evaluated control of the questions. With this information reports were generated showing the status of controls and the activities of Requirements Software.

A. *Questionnaire Preparation*

Figure 5 shows a screenshot of generation of a Software Requirements Questionnaire.

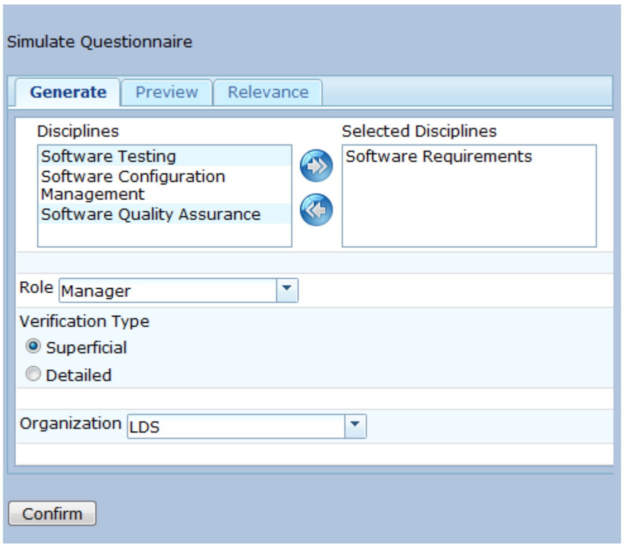

Figure 5. Screenshot, Generating Questionnaire of Software Requirements.

On the "Generate Questionnaire" use case, we generated two detailed questionnaires, one for the manager and one for the analysts. So, the following options were selected:

- Selected Disciplines: Software Requirements;
- Selected Role: Manager and after Analyst;
- Verification type: Detailed.
- Selected Company: The team's company was selected.

Clicking on the "Confirm" button, the system chooses the questions according to what was selected. The way it does this is filtering the questions by the disciplines that have been selected, and then the questions are filtered by the professionals' roles. Finally, the questions will be filtered by their verification type (superficial or detailed).

The expert can view the questions that will be part of the questionnaire on the "Preview" tab and add the relevance of the questions associated controls, tab "Relevance".

Note that the questions, activities and controls were previously registered in the knowledge base by experts in the field.

After that, the system looks for the employees who are already registered in the database and linked to the selected company. In this case, it has found the Analyst#1 and the Analyst#2, when the generated questionnaire was for Analysts, and the Manager#1 when the generated questionnaire was for manager.

Then the system saves the questionnaires in the database. These questionnaires will be answered later by employees when they access the tool. The employees are alerted by a notification email.

B. *Assessment and Information Collection*

When the company's employees access the tool by system login use case, they will be allowed to view the knowledge base and answer their specific questionnaires.

In this case study, the employees accessed the tool and answered 23 detailed questions, taking approximately the following times:

Analyst#1: 2 h 30 min
Analyst#2: 1 h 30 min
Manager#1: 1 h 45 min

At the end of the answers, they just clicked on "Submit" button and all the chosen answers were stored in the database.

When all the company's employees answer their questionnaires, the phase of Reports Generation begins.

C. *Reports Generation*

After all members of the development team answered the questionnaires, the responsible employee was alerted by email that the report generation was available. Thus, it is just log in the system and view the report.

The controls were counted according to their status and separate by disciplines and roles. For an overview, including results from both roles, the values of the results for analysts and managers are added. See the following tables Table II and Table III.

TABLE II
GENERAL RESUME CONTROL BY ACTIVITIES

Activity	Quantity of Controls				Indicator
	Imple-mented	Partially Imple-mented	Not Imple-mented	Total	Compliance Index
Demand Control	6	8	1	15	53,45%
Scope Definition	8	4	0	12	76,96%
Requirements Detailing	10	4	1	15	75,32%
Elicit Needs	4	2	1	7	71,62%
Requirements Management	3	3	3	9	33,33%
Change Control	4	5	3	12	41,95%
Requirements Review	3	2	4	9	50,62%
Requirements Approval	3	2	3	7	57,84%
Non-functional Requirements	1	1	7	9	17,78%
Maintenance and Evolution	7	7	1	15	60,61%
Total	49	38	23	110	57,47%

TABLE III
GENERAL RESUME - CONTROL

Control Status	Quantity
Implemented	49
Partially Implemented	38
Not Implemented	23
Not Answered	0
Total	110

For a better graphical representation, see the following figures Figure 6 and Figure 7.

The Figure 6 displays a graph (Pizza) of the total of evaluated controls, separating them by status.

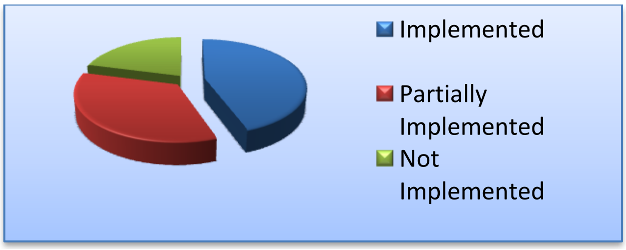

Figure 6. Controls' Status.

The Figure 7 shows a comparison between the activities, ordering by the worst compliance index.

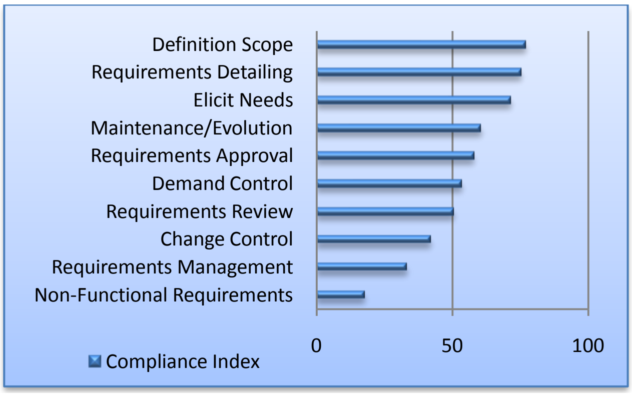

Figure 7. Activities Ordered by Compliance Index.

VI. Conclusion and Future Work

In the case study, it was found that the activities Non-Functional Requirements and Requirements Management have the lowest compliance index, and therefore they need a plan for improvement.

The main objective of the case study was to apply the tool in a real situation and verify its applicability. Thereby, the knowledge base was consistent and it was possible generate and evaluate the questionnaires.

For the generation of assessments in others disciplines or even in other levels of detail, it would be necessary to use Sábio to register the new knowledge and then generate new questionnaires automatically.

Then, with the tool, it is already possible to create, view, update and delete reference models, practices associated with these models, disciplines of software engineering, roles within the organization, and verification controls. It is also possible to relate questions to controls and generate questionnaires, as well as, interpret the answers chosen and indentify the riskiest disciplines, generating recommendations to improve the software process development.

Other use cases that are also implemented includes the "registration of the organization and user", "user login system", "roles and permissions for each kind of users in the system", and the "collection of the answers" from questionnaires that are being stored in database for evaluation.

With this work, we expected to contribute to the implementation of more efficient development processes, within quality standards. To improve software development, the field of Software Engineering is joining efforts to get better specifications, development and maintenance of systems, applying technologies, practices of project management and other disciplines. Moreover, all this demand reflects a side effect regarding the cost needed to be invested. This is because the development process requires the experience of the various methodologies of Software Engineering and also high qualified professionals.

This research, therefore, created a system that brings together in one environment the knowledge provided by various experts. A first motivation for developing the system was directly related to the possibility of organizing technical knowledge experienced through the use of customizable questionnaires. Another aspect that deserves attention is the aid provided by the tool, through reports, for emphasizing aspects of development that should be improved or met and the guides that explain how the improvements can be performed.

Acknowledgment

This work is partially supported by National Counsel of Technological and Scientific Development - Brazil (CNPq) under the project PROCOCO (620063/2008-4).

References

[1] M. B. Chrissis, M. Konrad, and S. Shrum. “CMMI: Guidelines for Process Integration and Product Improvement”. Boston: Addison-Wesley, 2003.

[2] A. Abran, J.W. Moore, P. Bourque, and R. Dupuis, “Guide to the Software Engineering Body of Knowledge (2004 Version)”. IEEE Computer Society Press, Los Alamitos, CA, 2004.

[3] Software Engineering Institute (SEI). “CMMI for Development, Version 1.2”. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, 2006b.

[4] ____. “ISO/IEC 15288:2008”. Available on-line at http://www.iso.org/iso/catalogue_detail?csnumber=43564/>. Accessed: 06/2010.

[5] K. Beck. “Extreme Programming Explained: Embrace Change”. Addison-Wesley, 1999.

[6] K. Schwaber, and M. Beedle “Agile software development with Scrum”, Prentice-Hall, 2002.

[7] D. M. Ahern, J. Armstrong, A. Clouse, J. R. Ferguson, W. Hayes, and K. E. Nidiffer. “CMMI SCAMPI Distilled: Appraisals for Process Improvement”. Addison-Wesley, 2005.

[8] SOFTEX. “MPS.BR - Melhoria de Processo do Software Brasileiro. Guia Geral”. Versão 1.1, 2006b.

[9] R. S. L. Espinha. “Uma Abordagem para a Avaliação de Processos de Desenvolvimento de Software Baseada em Risco e Conformidade”. 2007. Master's thesis - Pontifical Catholic University of Rio de Janeiro, Computer Science Department, Rio de Janeiro, 2007.

[10] R. A. Praes. “Ferramenta para Organização do Conhecimento e Avaliação de Práticas de Engenharia de Software (Sábio)”. 2009. Course Conclusion (Undergraduate) - Pontifical Catholic University of Rio de Janeiro, Computer Science Department, Rio de Janeiro, 2009.

[11] HS&M. CMM Quest v1.2. Available online at <http://www.cmm-quest.com/english/>. Accessed: 06/2010.

[12] ISD – Integrated System Diagnostics Incorporated. Displays Appraisal Wizard Tools Wizard Lite and to evaluate processes. Available online at <http://www.isd-inc.com/tools.appraisalWizard/>. Accessed: 06/2010.

[13] IME Toolkit, 2003. Interim Maturity Evaluation Toolkit, Management Information Systems. Available online at <http://www.man-infosystems. com/IMEtoolkit.htm/>. Accessed: 06/2010.

[14] M. Peralta, E. Diez, P. Britos, and R. G. Martinez. “Evaluación asistida de CMMI-SW”. Jornadas en Ingenieria de Sistemas Informaticos y de Computacion (JISIC 2004), 2004.

[15] F. J. Pino, F. Garcia, M. Piattini. “A support tool for rapid software process assessment”. Latin America Transactions, IEEE (Revista IEEE América Latina), vol. 5, 2007, p. 218-223.

[16] CMMI Models. Available on-line at http://www.sei.cmu.edu/cmmi/models/models.html/>. Accessed: 06/2010.

[17] ISO/IEC 12207:2002. Information technology - Software life cycle processes. International Organization for Standardization. Geneva. 2002.

 ISBN: 978-1-61208-107-6