IJCTA, 9(27), 2016, pp. 11-17
ISSN: 0974-5572

Software Measurement Linkage for CMMI Implementation

Prabu G.[1], Kannan N.[2], Kovalan A.[3] and Thaddeus S.[4]

ABSTRACT

Any decision regarding the maturity level of a process area in CMMI depends on the software metrics that are collected and analyzed. However, it is not easy to identify the measurement objectives for each process area and establish a measurement repository for an organization. Adopting a software measurement ontology (based on the existing standards for metrics) which includes all aspects of the metrics from its objective to realization, this paper presents a linkage framework to organize metrics in an effective manner within an organization. The framework provides facilities to define measurement objectives, identify the measurements, fix the measurement approach and apply it to the entities of the process or product. A prototype application is conceived to establish the proof-of-concept.

Keywords: CMMI, Software Measurement, Software Quality Assurance, Ontology, Process Assessment

1. INTRODUCTION

Historically, software quality assurance focused mainly on the final products such as deliverables, specifications and plans. A greater awareness has set in, that the quality of a software product is determined by the quality of its software development and maintenance processes. An effective approach to SQA is to monitor software activities continuously throughout development life cycle to ensure the quality of the delivered product. This is facilitated through many process frameworks such as CMMI, SPICE and ISO 9001 [1]. Software organizations apply one or more such frameworks to deliver products better, faster and cheaper. These process frameworks consider software measurement as a primary function to reach higher maturity levels. It plays a vital role to assess and institutionalize software process improvement programs.

Capability Maturity Model Integrated (CMMI) today is a globally preferred tool for process improvement. It depends on software measurement for each process area to validate if the goals of process areas are satisfied. Any organization that adopts CMMI must ensure that measurements are aligned to the business objectives to provide benefit, used regularly in order to justify the effort and cost, well defined in order for people to understand and compare and communicated in an unbiased manner. This requires that each organization has a meaningful mapping of software measurement to CMMI implementation. Using distinct ontologies for CMMI and Software Measurement, goals of CMMI process areas can be mapped to information objective of software measurement ontology. Software Measurement ontology handles separately all aspects related to measurement such as the objective of measurement, method of measurement, metrics, entities, analysis model and the final interpretation. This provides a measurement framework for an organization to define its own measurement system which can be independently managed and which can also be applied for CMMI implementation.

Ontologies in Computer Science represent formal, explicit specification of a shared conceptualization [2]. Ontologies provide a common understanding of a domain that can be communicated between people and application systems. Ontology presents a system of concepts (or classes) and their relations (or properties), where concepts are defined and interpreted in a formal manner.

[1] Associate Professor, Jayaram College of Eng. and Tech., *Email: vgprabhu.samy@gmail.com*

[2&4] Principal, Jayaram College of Eng. and Tech., Don Bosco College Yelagiri, *Emails: principal@jayaramcet.edu.in, thad@boscoits.com*

[3] Assistant Professor, Periyar Maniammai University, *Email: kovapmu@gmail.com*

Many ontology representation languages are proposed based on knowledge representation schemes from first-order logic to frame-based structures. In the context of Semantic Web, RDF (Resource Description Framework) with OWL (Web Ontology Language) is widely accepted as knowledge representation languages [3]. OWL is a consolidation of its preceding languages. OWL provides a rich set of constructs to define classes using class axioms, logical operations and property restrictions. Classes are linked to one another using ontology properties namely object, data and annotation properties. Object properties define the relationship between concepts. Data properties provide attributes to individual members and annotation properties describe the characteristics of an instance of a class. Instances of ontology classes are called as the individuals that form the class. When formal axioms are added to ontology concepts, they facilitate automated inferencing of new knowledge from the existing knowledge. Ontologies serve as the backbone of our approach where the ontology for CMMI is linked to ontology for software measurement. Ontology merging technique is used to connect both as a 'view'.

2. LITERATURE REVIEW

Information needs, measurement constructs, and measurement procedures are combined into a measurement plan. Execution of the measurement plan produces the information products that respond to the project information needs [4]. The most important aim of software engineering is to improve software productivity and quality of software product and further reduce the cost of software and time using engineering and management techniques [5].

Although the process of measuring in software engineering has already been standardized in the ISO/IEC 15939 standard, where activities related to identifying, creating, and evaluating of measures are described, a standard tailored measurement systems are desired [6].

Semantic interoperability needs to be integrated to support the software measurement process, such as jointly using tools for project management, quality assurance and process management [7].

3. CMMI ONTOLOGY

CMMI is a single, integrated and agile framework for guiding and appraising improvement activities in software engineering, systems engineering, integrated product and process development and supplier sourcing. CMMI

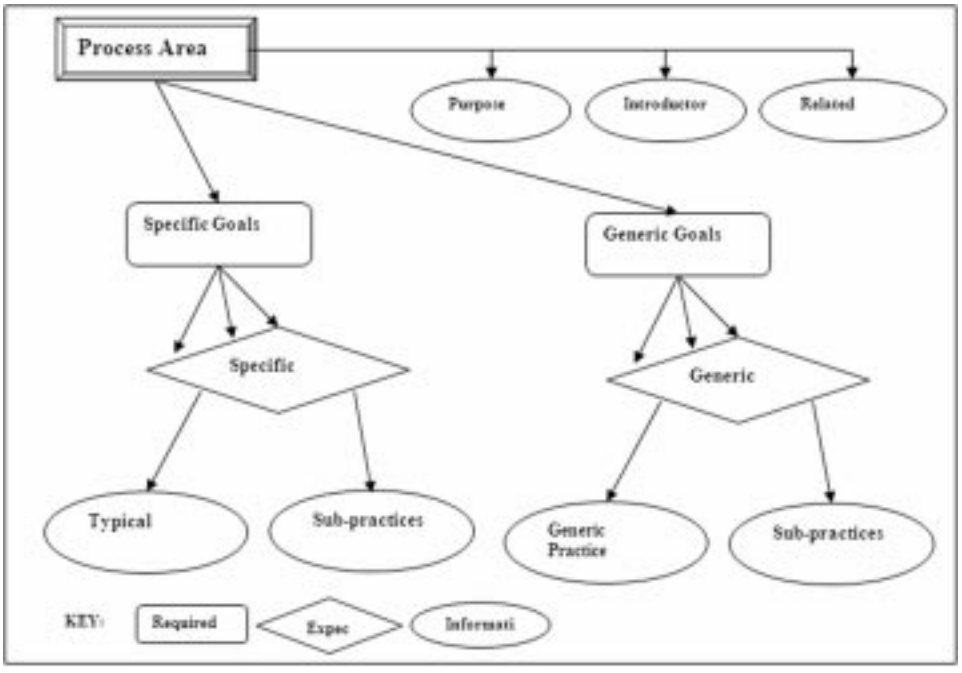

Figure 1. CMMI v1.2 Model Components

Product Suite provides the resources to evaluate the content of an organization's processes and to improve its process performance [8, 10]. The main components of CMMI Model v1.2 (as shown in Figure 1) are process areas, goals, practices and work-products.

In the context of CMMI model, concepts of CMMI are expressed in OWL-DL using its class axioms (inclusion and equivalence definitions), class descriptions (with constructors such as intersection, union, complement and disjoint) and property restrictions. OWL-DL is an offshoot of the family of Description Logic, which is a derivative of First Order Predicate Logic, offering high expressivity with sound and decidable reasoning facility. Thus, a semantic conceptualization of CMMI model is realized using OWL.

The concepts and structure of any ontology is decided much by the purpose it has to serve. The required and expected components of CMMI listed above, process areas, goals, and practices, form automatically the core classes of CMMI ontology. Process areas are conceptualized under four categories of Engineering, Support, Process and Project. Each process area is defined as ontology classes with the goals to be satisfied as the restricted conditions under one of the four groups. Figure 2 gives the scheme of the developed ontology model. All process areas under engineering are alone shown in the figure. The specific and generic goals needed to obtain maturity levels from 2 to 5 are incorporated as subclasses. All goals are grouped as generic or specific concepts. To define specific goals, concerning each process, separate process-related goal concepts are introduced. The actual specifications of various goals, be it generic or specific related to various concepts are listed as individuals.

The ontology properties are used to link various process concepts in a meaningful and logical manner as per domain understanding. Using the property <consistsof>, each maturity level class is defined. For instance maturity level 2 (ML2) is defined as consisting of process areas namely Requirement Management (RM), Configuration Management (CM), Project Planning (PP), Project Monitoring and Control (PMC), Measurement and Analysis (MA), Supplier Agreement Management (SAM) and Process and Product Quality Assurance (PPQA). <satisfiedbyGG> and <satisfiedbySG> are used to connect generic and specific goals with the respective process areas. Specific and respective practices are linked to the respective goals using the property <achievedBy>. The work products that are produced as a result of specific or generic practices are linked using two properties namely <hasDirectEvidence> and <hasIndirectEvidence>. <fulfillsGoal> relates the practices with the corresponding goals as an inverse of the properties <satisfiedbyGG> and <satisfiedbySG>.

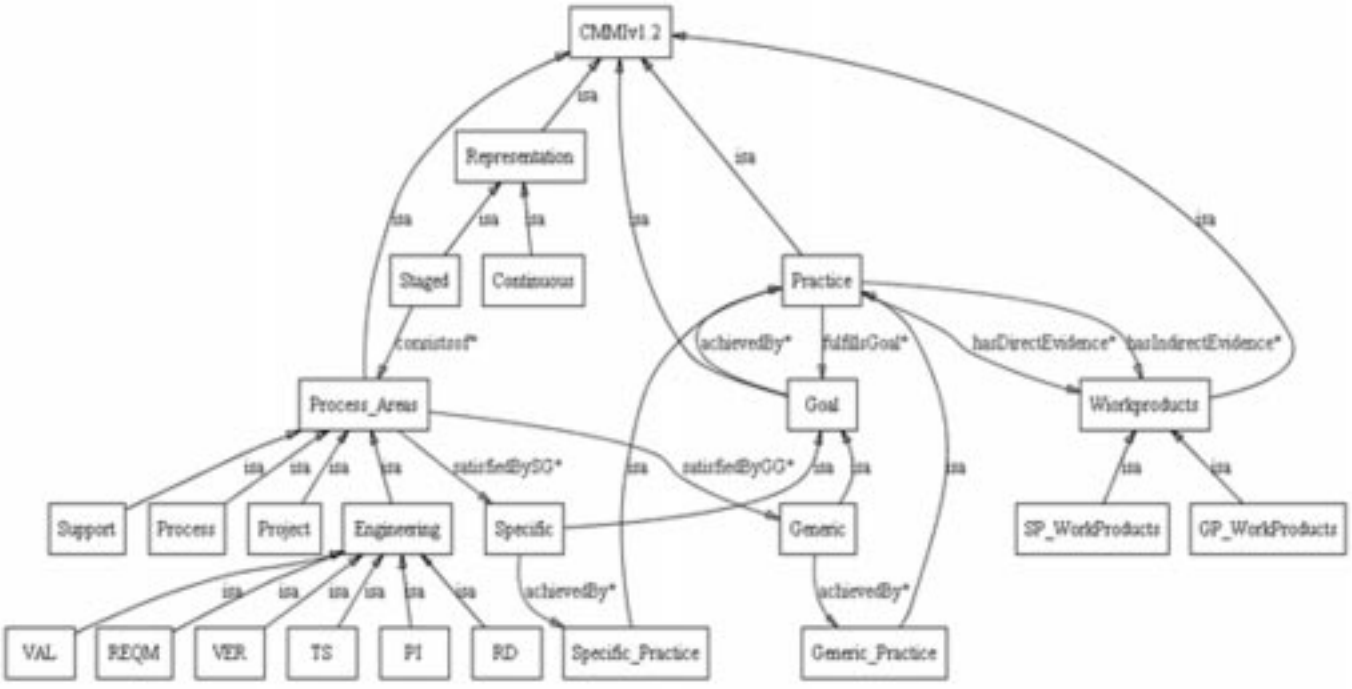

Figure 2: Ontology Model of CMMI v1.2

4. SOFTWARE MEASUREMENT ONTOLOGY

ISO 15939 defines an approach to software measurement. Basing on this, an ontology for software measurement (SMO) is proposed by Alarcos Research Group [10]. It is organized around four main sub-ontologies: Software Measurement characterization and objectives, to establish the context and goals of the measurement; Software Measures, to clarify the terminology involved in the measures definition; Measurement Approaches, to describe the different ways of obtaining the measurement results for the measures' and Measurement, which includes the concepts related to performing the measurement process as shown in Figure 3.

The "Software Measurement Characterization and Objectives" sub-ontology includes the concepts required to establish the scope and objectives of the software measurement process. The main goal of a software measurement process is to satisfy certain information needs by identifying the entities and the attributes of these entities. Attributes and information needs are related through measurable concepts. The "Software Measures" sub-ontology aims at establishing and clarifying the key elements in the definition of a software measure. A measure relates a defined measurement approach and a measurement scale. Most measures may or may not be expressed in a unit of measurement, and can be defined for more than one attribute. Three kinds of measures are distinguished: Base Measures, Derived Measures and Indicators. The "Measurement Approaches" sub-ontology introduces the concept of measurement approach to generalize the different "approaches" used by the three kinds of measures for obtaining their respective measurement results. A base measure applies a measurement method. A derived measure uses a measurement function. Finally, an indicator uses an analysis model to obtain a measurement result that satisfies an information need. The "Measurement" sub-ontology establishes the terminology related to the act of measuring software. A measurement is a set of operations having the object of determining the value of a measurement result, for a given attribute of an entity, using a measurement approach. Measurement results are obtained as the result of performing measurements.

5. PROPOSED LINKAGE FRAMEWORK

Our developed CMMI ontology with the adopted SMO will have no significance if it does not find a practical application. While CMMI ontology and SMO remain independent entities, a merging of these is done using a bridge ontology. An ontology merging algorithm known as PROMPT [11] is used for this. A bridge axiom is used to connect process area goals with information need of SMO. This method is preferred, instead of creating a new, target ontology, so that both CMMI and SMO can be independent and be treated separately. The bridge ontology merely provides a 'view' by merging both ontologies based on individuals in classes.

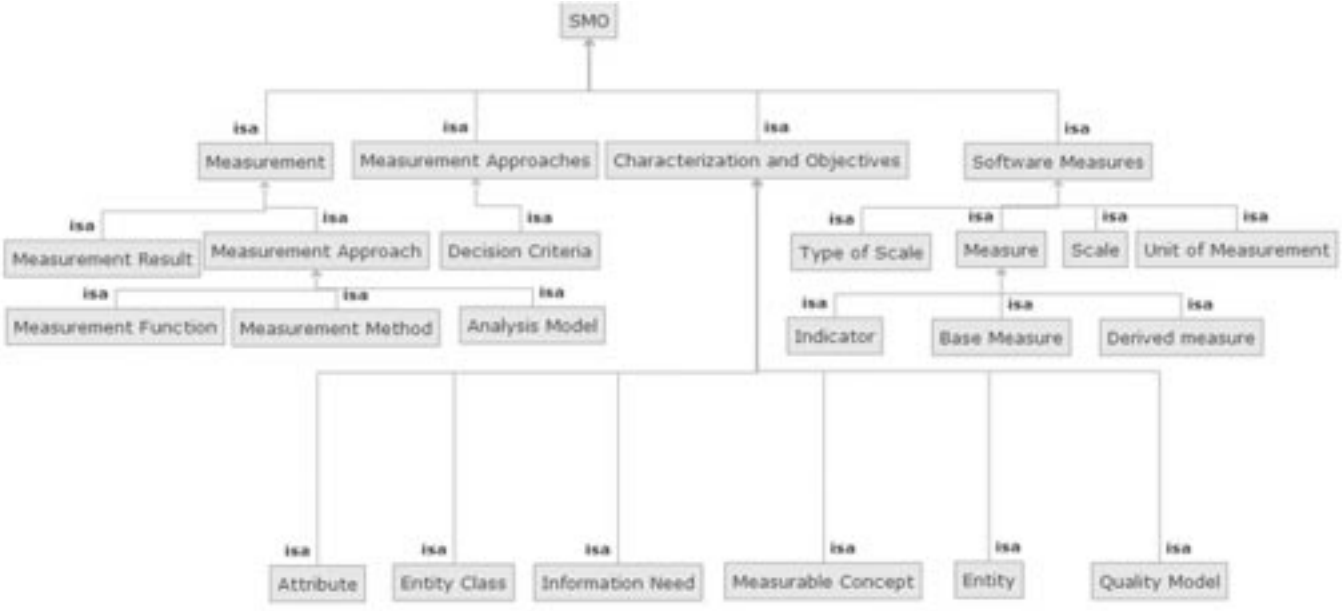

Figure 3: SMO Model

The measurement need of Anadocs Software Company in Chennai which is in the process of obtaining CMMI level 3 is taken for our case study. For each goal of CMMI process area, the corresponding information need for measurement is fixed. According to the information need, the possible measurements are identified and linked through our prototype tool. The tool provides an interface for the end users to update SMO and set its usage for any CMMI goal. Further, applications can be developed to automate the process of Goal Satisfaction based on the measurement obtained. Table 1 presents a few measurements defined for the organization with regard to project management which are stored as instances of SMO.

Table 2 gives partial mapping between CMMI goals of different process areas and Information Need of SMO. The existence of SMO gives conceptual clarity for measurement practitioners and quality assurance team.

Table 1

Information Need	*Attributes*	*Base Measure*	*Derived Measure*	*Indicator*
How good is the Effort Estimation planning in a project?	Effort Estimation	Estimated Hours Actual Hours	Effort Variance (%)	1. If Effort Variance Diff > 15%, then the Project Estimation Planning is NEGATIVE. 2. If Effort Variance Diff < 5%, then the Project Estimation Planning is POSITIVE.
Has the project been on schedule throughout?	Task Duration	No. of Tasks Planned No. of Tasks Completed	Schedule Variance (%) (in terms of Tasks)	1. If Schedule Variance Diff > 15%, then the Planned Schedule is NEGATIVE. 2. If Schedule Variance Diff < 5%, then the Planned Schedule is POSITIVE.
Have QA activities been efficient in defect removal?	Defects	No. of Defects before delivery No. of Defects after delivery	Defect Removal Efficiency (DRE) (%)	1. If DRE < 95%, then the performance of the project's software process is NEGATIVE. 2. If DRE > 95%, then the performance of the project's software process is POSITIVE.
What is the average effort spent for the past six months projects for different activities?	Effort Estimation	This metric indicates the effort spent for the past six months projects per activity • Requirement • Design • Coding • Testing • Support/ Implementation • Maintenance	Effort Distribution (%)	Effort Distribution / Percentage / Activities
How critical the defects are?	Defects	No. of Defects per Defect Type wise	Defect Distribution - Testing (%)	

Table 2

Process Area	*Goal*	*Information Need*
PP, PMC, IPM	PP SG 1 - Establish Estimates PMC SG2 - Manage Corrective Action to Closure IPM SG1- Use the Project's Defined Process	How good is the Effort Estimation planning in a project?
PP, PMC, IPM	PP SG 1 - Establish Estimates PMC SG2 - Manage Corrective Action to Closure IPM SG1- Use the Project's Defined Process	Has the project been on schedule throughout?
REQM, RD,TS, PI, VER,VAL	GP 2.8 Monitor and Control the Process	Have QA activities been efficient in defect removal?
PP, PMC, IPM	PP SG 1 - Establish Estimates PMC SG2 - Manage Corrective Action to Closure IPM SG1- Use the Project's Defined Process	What is the average effort spent for the past six months projects for different activities?
REQM, RD,TS, PI, VER,VAL	GP 2.8 Monitor and Control the Process	How critical the defects are?

The team can improve the set of measurements in an organization and use them to satisfy multiple needs. Like CMMI, any other process framework can also be linked to the measurement system of the organization. This framework will facilitate easier, reliable and practical measurement practices in any software organization.

6. CONCLUSIONS

CMMI ontology, presented in this paper, was developed using Protégé as the ontology editor [8]. The authors are ready to share the ontology with any reader who is interested in experimenting it and make it more relevant to software engineering community. Based on many iterations and experiment with multiple options, this ontology has been developed. The validity of the ontology has to be shown by its output.

SMO has been adapted from the work of Alarcos Research group. They provide it as a set of guidelines for measurement practitioners who may be confused by the terminology differences and conflicts in the existing standards and proposals. Further, it provides a cohesive core set of concepts and terms over which their existing standards could be integrated or new ones built. Our contribution in this paper is to take SMO beyond being a pure reference model to an application model. It has been applied and linked to CMMI concepts to facilitate the automation of process appraisal. A software tool is being developed to implement the merging and updating ontologies in a user-friendly manner and to automate the process of CMMI appraisal. Specific axioms can be integrated so that the rating of processes can be done for assessment and certification of process framework. This can emerge as an extension of our work after testing the ontology with software projects of multifarious organizations.

REFERENCES

[1] Silvia T Acuna, Angelica de Antonio, Xavier Ferre, and Martia Lopez, "The Software Process: Modeling, Evaluation and Improvement", *Handbook of Software Engineering & Knowledge Engineering*, **1** Fundamentals, World Scientific Publishing, 200-210, 2001.

[2] Gruber T.R., "A Translation Approach to Portable Ontology Specifications", *Knowledge Acquisition*, **5**, 199-220, 1993.

[3] Carnegie Mellon University, Software Engineering Institute, "CMMI for Development Version 1.2", *CMMI-DEV/CMU/SEI-2006-TR-008*, August 2006.

[4] Srinivasan K P, "Unique Fundamentals Of Software Measurement And Software Metrics In Software Engineering", *International Journal of Computer Science & Information Technology (IJCSIT)* **7(4)**, August 2015.

[5] Buse, Raymond PL, and Thomas Zimmermann. "Information needs for software development analytics" *Proceedings of the 34th international conference on software engineering*, IEEE Press, 2012.

[6] Mrinal Singh Rawat, Arpita Mittal and Sanjay Kumar, "Survey on Impact of Software Metrics on Software Quality," *International Journal of Advanced Computer Science and Applications*, **3(1)**, 2012

[7] Rakesh.L , Dr.Manoranjan Kumar Singh ,and Dr.Gunaseelan Devaraj , "Software Metrics: Some degree of Software Measuremen and Analysis," *International Journal of Computer Science and Information Security*, **8(2)**, 2010

[8] Horrocks, I., Patel-Schneider, P.F. and van Harmelen, F, "From SHIQ and RDF to OWL: The making of a Web Ontology Language", *Journal of Web Semantics*, **1(1)**, 7–26, 2003.

[9] Mary Beth Chrissis, Mike Knorad and Sandy Shrum, "CMMI Guidelines for Process Integration and Product Improvement", *Pearson Education*, Second Edition, 2006.

[10] Garcia, F., Bertoa, M., Calero, C., Vallecillo, A., Ruiz, F., Piattini, M. and Genero, M, "Towards a consistent terminology for software measurement. Information and Software Technology," *Information and Software Technology*, 631-644, 2006.

[11] Noy NF and Musen MA, "PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment" Proceedings of *American Association for Artificial Intelligence -2000 (AAAI-00)*, 2000.

[12] Gennari, J., et al., "The Evolution of Protégé: An Environment for Knowledge-Based Systems Development", *International Journal of Human-Computer Studies*, **58(1)**, 89–123, 2003.

[13] ISO/IEC. ISO/IEC 12207: System and software engineering – Software life-cycle processes, Second edition, 2008.

[14] Software Engineering Institute. CMMI for Development, Version 1.3, Technical Report CMU/SEI-2010-TR-033, 2010.

[15] ISO/IEC. ISO/IEC 15939: Software Engineering – Software Measurement Process, Second edition, 2007.

[16] Mcgarry, J., Card, D., Jones, C., Layman, B., Clark, E., Dean, J. and Hall, F. "Practical Software Measurement: Objetive Information for Decision Makers", Addison Wesley, 2002.

[17] Monalessa Perini Barcellos, Ricardo de Almeida Falbo, "A Software Measurement Task Ontology," *Association of Computing Machinery ACM*, ISSN: 978-1-4503-1656-9, 2013.

[18] "A Guide to the Software Engineering Body of Knowledge SWEBOK", IEEE Computer Society: Los Alamitos, CA, http://www.swebok.org

[19] Calhau, R.F., and Falbo, R.A. "A Configuration Management task ontology for semantic integration" *Proceedings of the 27th Annual ACM Symposium on Applied Computing (ACMSAC´2012)*, 348 – 353, 2012.

[20] N. E. Fenton and S. L. Pfleeger, "Software Metrics: A Rigorous and Practical Approach," *International Thomson Publishing*, II Edition, London, 1996.

[21] Tim Berners-Lee, "Future Of The Web" http://dig.csail.mit.edu/2007/03/01-ushouse-future-of-the-web, 2007.

[22] Nachiappan Nagappan, Brendan Murphy, and Victor Basili, "The Influence of Organizational Structure On Software Quality: An Empirical Case Study", *Association of Computing Machines ACM*, 978-1-60558-079-1, 2008.

[23] Staron, Miroslaw, et al. "Developing measurement systems: an industrial case study." *Journal of Software Maintenance and Evolution: Research and Practice*, 89-107, 2011.