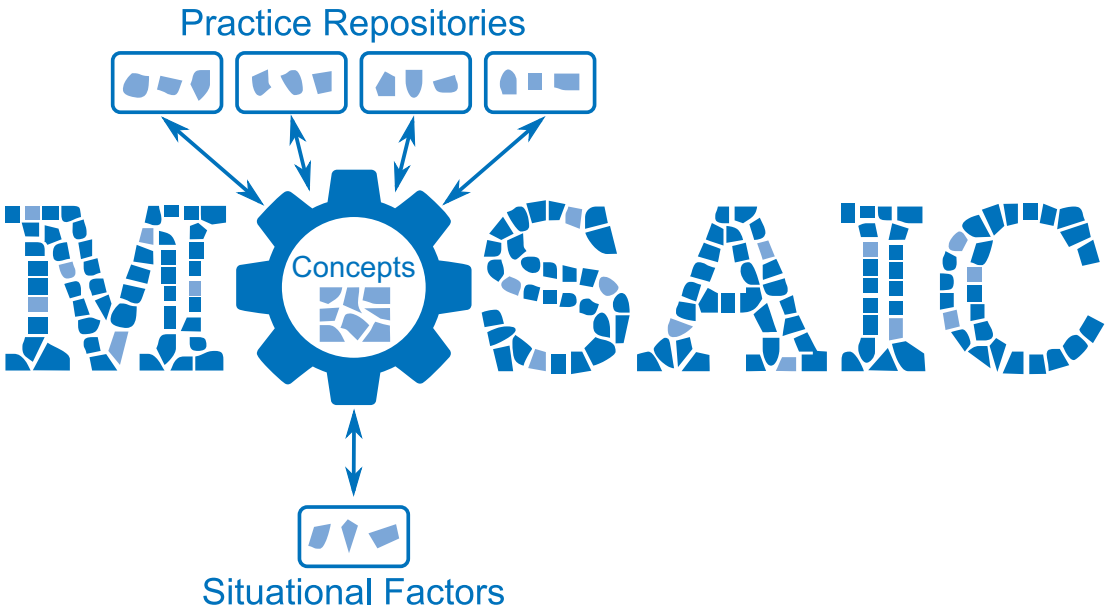
RWTH AACHEN

SE SOFTWARE ENGINEERING

Simona C. Jeners

Model-supported Process Adoption and Assessment in the Context of Multiple Practice Repositories



Aachener Informatik-Berichte,
Software Engineering

Band 21

Hrsg: Prof. Dr. rer. nat. Bernhard Rumpe

Model-supported Process Adoption and Assessment in the Context of Multiple Practice Repositories

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades einer Doktorin der Naturwissenschaften oder einer Doktorin der Ingenieurwissenschaften genehmigte Dissertation

vorgelegt von

Simona Cristina Jeners (geb. Pricope), M. Sc.

aus Iaşi, Rumänien

Berichter: Univ.-Prof. Dr. rer. nat. Horst Lichter

Prof. Luigi Buglione, PhD

Tag der mündlichen Prüfung: 17. Dezember 2014

[Jen15] S. C. Jeners:
Model-supported Process Adoption and Assessment in the Context of Multiple Practice Repositories.
Shaker Verlag, ISBN 978-3-8440-3481-3. Aachener Informatik-Berichte, Software Engineering, Band 21. 2015.
www.se-rwth.de/publications/

Summary

Although software products exist for more than 60 years, their successful development within a software project is still a challenge. A reason for this high failure rate might be poor software processes and the lack of systematic software process improvement. Considerable attention has been given to software process improvement in the last years and thus, practice repositories, such as CMMI-DEV, SPICE, COBIT, ITIL, have been defined. These are collections of best practices that describe activities which have proven themselves as guidelines for the improvement of software processes in organizations.

Organizations use the practice repositories by adopting them or by performing assessments based on them. Moreover, many organizations aim to use multiple practice repositories to benefit from synergy effects and to achieve a higher effectiveness in the software process improvement. However, there are several challenges when using multiple practice repositories. These repositories have to be addressed in a coordinated and systematic way to benefit from them correspondingly.

MOSAIC, a model based approach, is proposed to support the usage of multiple practice repositories and achieve an effective and efficient adoption and assessment based on an integration of these repositories with the software project context. This integration allows various analysis activities to be automatically performed on the integrated models. For this purpose, the structure and terminology of the various practice repositories is normalized, as well as the software project context is modelled by defining situational factors that describe it.

An integration on a common structure and terminology is possible at a conceptual level. The concepts and their similarity relations are extracted from the practice repositories and saved in a central model. This central model relates the various repositories with each other and the repositories with the situational factors and thus, with the software project context. This approach is flexible and allows the integration of practice repositories from different software areas. Therefore, based on the purposes and needs of organizations, the integration can be extended over time into a broader process consideration. Furthermore, specific situational factors that do not describe only the software project settings but also other working contexts can also be added to satisfy specific needs of organizations.

This integration allows various analysis activities on the created models. Consequently, best suited practices from multiple practice repositories can be automatically selected, similar practices and the dependencies between them can be automatically identified. All these analysis activities are based on various metrics that measure the support degree of the practices for the software project context, their similarity and dependency degree respectively.

All these analysis automations can be utilized by organizations when attempting to make key process decisions in their software process improvement. Various applications of MOSAIC for organizations that work with multiple practice repositories are listed and described in this work. Hence, a guidance for software process improvement initiatives of organizations is provided to allow an effective and efficient adoption of such repositories and assessments according to these repositories.

Acknowledgement

A long journey comes to an end. With the support of the family, supervisors, colleagues and friends, I had a remarkable and wonderful journey that gave me the chance to widen my knowledge and wisdom. Therefore, I would like to thank all these people that contributed to this journey.

First of all, I would like to thank Professor Dr. Horst Lichter that continuously provided me constructive feedback and an environment which encouraged me to enjoy my research. Thanks for your patience and your support! I would also like to thank my second supervisor, Professor Luigi Buglione, PhD, for the remarks and suggestions during the last phase of this work.

I am thankful to the many experts from different IT organizations who contributed to the improvement and evaluation of the approach presented in this work. Discussions, interviews or experiments with my former colleagues from Kugler Maag CIE, especially with Dr. Ute Streubel, Frank Sazama and Dr. Klaus Hörmann, enhanced my knowledge in the software process improvement and gave me the opportunity to recognize the weaknesses of this approach and improve them. Furthermore, I would like to thank two organizations, Generali Informatik Services and ITERGO that offered me the possibility to gain an insight into the various challenges of organizations working with multiple practice repositories. The following ITERGO colleagues supported me during the evaluation of this work: Burkhard Bujotzek, Martina Funk, Rolf Schmitz and Martina Weber. Finally, I would like to thank my colleagues from Ireland, especially Professor Dr. Rory O'Connor and Dr. Paul Clarke for providing valuable input for the improvement and evaluation of the approach presented in this work.

Nevertheless, I would like to thank my colleagues from my research department who supported this work and provided me valuable input. They were actively involved in discussions and workshops, listened carefully to my ideas, put the right questions at the right time and provided me suggestions and solutions that led to a more efficient and effective development of my approach. They did not only provide a technical environment where ideas could be discussed and improved, but also a friendly and cooperative atmosphere. I also mention the valuable contribution of my students, especially of Elena Pyatkova, Ana Dragomir and Togrul Mageramov. They worked intensively and achieve promising results for the presented approach during their master theses.

Finally, my family plays an important role for the development of this work as they emotionally supported me during all these last years, praised me in good times and encouraged me in difficult ones. In particular, I would like to thank Nils Jeners, my husband, for his patience and love.

Table of Contents

1 INTRODUCTION 1

1.1 Usage of Practice Repositories 3
1.2 Experiences with Multiple Practice Repositories 4
1.3 Challenges with Multiple Practice Repositories 4
1.4 MOSAIC Approach 8
1.5 MOSAIC Challenges, Research Questions and Working Fields 9
1.6 Thesis Structure 11

2 OVERVIEW 13

2.1 Approach 13
2.2 Example Scenario 15
2.3 Analysis Activities and Metrics 16
2.4 Models and Modeling Activities 23
2.5 MOSAIC Toolbox 32
2.6 Analyzer and Modeler Roles 33
2.7 Summary 34

3 META-MODELS 37

3.1 Integrated Structure Meta-Model 37
3.2 Integrated Concept Meta-Model 40
3.3 Situational Factors Meta-Model 43
3.4 Summary 46

4 MODELING ACTIVITIES 47

4.1 Running Example 48
4.2 Structure Normalization 48
4.3 Terminology Normalization 50
4.4 Integration of the Software Project Context 64
4.5 Summary 68

5 ANALYSIS ACTIVITIES AND METRICS 69

5.1 Running Example 69
5.2 Measurement Theory 69
5.3 MOSAIC Metrics 70
5.4 Selection of ISM Practices 72
5.5 Identification of Similar ISM Practices 79
5.6 Identification of ISM Practice Dependencies 108
5.7 Summary 114

6 MOSAIC TOOLBOX 115

6.1 Functional Requirements 115
6.2 Overview of the Tools within the MOSAIC Toolbox 116
6.3 Architecture 118
6.4 Implementation 123
6.5 Summary 135

7 APPLICATIONS 137

7.1 Overview 138
7.2 Process Profile of an Organization 141
7.3 Value of a Reference PR 142
7.4 New PRs 143
7.5 Tailoring Instrument for Software Projects 144
7.6 Repository of Multiple PRs 145
7.7 Avoid Redundancies 148
7.8 Avoid Inconsistencies 149
7.9 Provide Helpful Information for Adoption 150
7.10 Maintenance of Internal Process PRs 151
7.11 Compliance to Reference PRs 152
7.12 Efficient Assessments 153
7.13 Non-redundant and Non-conflicting Requirements 154
7.14 Summary 155

8 EVALUATION 159

8.1 Types of Evaluation 159
8.2 Subjects Profile 161
8.3 Experiments 162
8.4 Case and Field Studies 167
8.5 MOSAIC Toolbox Evaluation 195
8.6 Summary 203

9 CONCLUSIONS 207

9.1 Contributions 207
9.2 Summary 209

10 APPENDICES 213

10.1 MOSAIC Toolbox requirements 213
10.2 MOSAIC Toolbox Handbook 223

11 ACRONYMS 239

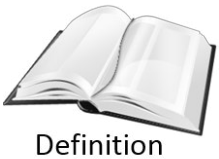
12 DICTIONARY 241

13 BIBLIOGRAPHY 245

1 Introduction

Nowadays, software becomes more and more ubiquitous in our professional and everyday life. Software products exist in different domains, for example in banking, insurance, automotive or health care industry. Although software products exist for more than 60 years, their successful development within a software project is still a challenge. For example, the Standish Group[1] regularly reports that the failure rate of software projects is still high: 61% of software projects do not meet their deadlines nor achieve the requested quality or are cancelled [The Standish Group International 2013]. Furthermore, the U.S. Department of Defense, which contracts numerous organizations to develop software products, reports that 40% to 60% of the project budget is spent on rework [Dekkers 2013]. It seems that software projects still have serious problems in their execution. A reason for this high failure rate might be the poor software processes and the lack of a systematic software process improvement [Paulk 1994]. Poor software processes might lead to low application quality as the software process quality influences the application quality [Paulk 2010]. With software processes we do not only refer to the processes for the software development, but also to the processes for other related software areas, such as software operation, IT governance or portfolio management. The different software areas are strongly related and thus, the success of the software development also depends on the success of the processes for these other areas.

Considerable attention has been given to software process improvement in the last years. The software engineering research provides a wide variety of best practices.



Definition

A **best practice** (in short practice) describes an activity which has proven itself as a guideline for the improvement of software processes in an organization.

Various acknowledged software development professionals, such as Humphrey [Humphrey 1988], have long recommended collections of such practices to be used for the software process improvement in organizations. Such collections can be used to systematically improve the quality of software processes. The process quality influences not only the software application quality, but also reduces time-to-market and production costs [Dyba 2005]. 148 analyses demonstrate the impact of the various software process improvement initiatives on quality, cost and schedule [Unterkalmsteiner et al. 2012]. Hence, these collections may support the software projects to achieve their goals.

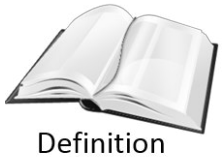
There exist various collections of best practices: capability models, standards, libraries, frameworks or process models. Here are some examples:

- Capability Models within the CMMI constellations: CMMI-DEV v1.3 [CMMI Product Team 2010a] or CMMI-SVC 1.3 [CMMI Product Team 2010b]

[1] Standish Group is a "group of highly dedicated professionals with years of practical experience in assessing risk, cost, return and value for IT Investments" (Standish Group – About).

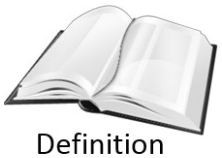
- ISO Standards: SPICE [ISO/IEC 15504-x 2010], ISO/IEC 12207 [ISO/IEC 12207:2008 2008], IEC 61508 ed. 1 [DKE Gremium 914 1999]
- Libraries: ITIL v3 [AXELOS 2011]
- Frameworks: COBIT v5 [ISACA 2011a] or VAL IT v2 [ISACA 2008]
- Process Models: V-Model XT [Höhn 2008], SCRUM [Sutherland and Schwaber 2013] or XP [Beck 2000]

As listed above, there is no standardized name for these collections. Different software process improvement approaches acknowledge this fact. For example, they are called quality standards, quality assurance methods, improvement frameworks [Paulk 1994], software process improvement frameworks [Halvorsen and Conradi 2001], improvement technologies, models [Siviy et al. 2008a] or process improvement models [Ferreira et al. 2010]. We call them all practice repositories as a repository commonly refers to a location where goods (best practices) are safely stored for the future use. Therefore, we give the following definition of a practice repository:


Definition

A **practice repository** (in short PR) is a collection of practices defined as being based on the experience and knowledge of many practitioners over the years in order to be used for the improvement of software processes in an organization. A practice repository does not only contain practices, but also other elements that are related to these practices (e.g. processes or activities are such elements).

Based on their level of detail and on their applicability, there are different types of PRs that contain practices to be considered for the software process improvement:


Definition

A **reference PR** is used as a guideline for the software process improvement of organizations. For example, CMMI-DEV, COBIT or ISO/IEC 12207 are reference PRs.

A **process PR** refers to a process model and is more concrete than a reference PR because it defines not only practices, but also gives additional information about how to adopt these practices. It can be used as a guideline, but it can also be directly applied to describe the software processes of an organization. For example, the V-Model XT is such a process PR.

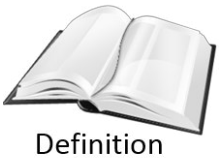
An **internal process PR** is a special process PR and refers to the internal software processes of an organization. This PR defines activities to be used as guidelines for the improvement of software processes in an organization, e.g. in software projects. Hence, it defines the practices of this organization and thus, it is a PR.

The term **PR** does not necessarily refer to all elements of a certain PR, but can also refer to a subset of elements of this certain PR that are selected for the software process improvement in an organization.

There exist numerous PRs that are defined during the last years. For example, there exist 315 standards and 52 capability models [Moore 1999; von Wangenheim et al. 2010]. As these PRs differ to some extent, there are taxonomies which categorize them to better grasp their purpose and differences [Minnich 2002; Mora et al. 2008; Siviy et al. 2008c]. For example, PRs are categorized according to their abstractness (abstract vs. detailed PRs, e.g. CMMI-SVC vs. ITIL), according to the software area (software development vs. software operation, e.g. CMMI-DEV vs. CMMI-SVC) or according to their methodology (agile vs traditional, e.g. ISO/IEC 12207 vs. SCRUM). All these PRs can be used for the software process improvement of organizations.

1.1 Usage of Practice Repositories

The usage of PRs for the software process improvement in an organization refers to the adoption of PRs and eventually to the assessment based on PRs. To get a common sense of what adoption and assessment mean in the context of this thesis, we define these terms in advance.



An **adoption of a PR** is the process of following this PR by an organization in order to improve its software processes. The adoption does not only mean the organization-wide process improvement to define an internal process PR, but also the adoption of this PR in software projects.

An **assessment based on a PR** is the evaluation process of software processes of an organization or part of an organization according to a PR to identify deviations from this PR and thus, to identify improvement potential for the software processes. If such an assessment is successful, we say that the organization is PR compliant.

For such an adoption and assessment, an organization needs to decide which PRs have to be used. An organization can use one or more PRs for their process improvement.

Many organizations decide to use multiple PRs to increase their competitive strength on the market, to fulfill the requirements of a customer or certain law regulations. For example, in the automotive industry some suppliers work with CMMI-DEV and SPICE to fulfill the requirements of different OEMs (Original Equipment Manufacturers).

Furthermore, organizations do not take into consideration only one software area, as their aim is to address the improvement of the software processes in the entire organization. For example, VAL-IT helps an organization to systematically address its portfolio management, COBIT to manage the IT governance, ITIL to manage the software operation (or service operation), and CMMI-DEV or SPICE to manage the software development.

Finally, organizations also aim to not entirely adopt all the elements of a PR, but to select only some of its elements that are the most relevant for their needs. Institutions that define PRs acknowledge this fact as well. For example, based on the needs of an organization, the continuous representation of CMMI proposes to select best suited process areas for the process improvement until a certain capability of the selected process area is achieved. This selection allows organizations to combine the elements of multiple PRs to address their needs. Consequently, elements of multiple PRs can be selected to support the software process improvement in an organization.

Hence, there is no PR that is the perfect solution for an organization and addresses all its needs. According to the contingency theory, the optimal course of action for an organization is to be contingent with its needs and settings [Morgan 2006]. Boehm and Turner suggest that when it comes to software processes, it seems likely that the claim "*one size fits all*" is, in fact, a myth [Boehm and Turner 2004:7]. Consequently, the organizations' diversity has to be sustained by individual approaches that combine multiple PRs for the adoption and assessment [Siviy et al. 2008a].

1.2 Experiences with Multiple Practice Repositories

The adoption of multiple PRs bring several advantages to organizations, e.g. higher effectiveness of process improvement, efficiency increase, cost reductions or the enhancement in the achievement of project goals [Siviy et al. 2008a]. Several organizations show some positive experiences with the adoption of multiple PRs, e.g. in Lockheed Martin IS&GS, Northrop Grumman Mission Systems or Wipro [Siviy et al. 2008a]. These are big organizations. Such organizations mostly show positive experiences with traditional PRs [Gibson et al. 2006; Herbsleb and Goldenson 1996]. In small organizations there are some positive experiences with the traditional PRs [Cater-Steel and Rout 2008; Cepeda et al. 2008; Habra et al. 2008], but these PRs are often perceived as too heavy and bureaucratic [Fowler and Highsmith 2001]. Therefore, small organizations tend to use agile PRs because of their benefits [Conboy 2009; Dybå and Dingsøyr 2008; Hossain et al. 2009]. However, this is not a general rule as there exist big organizations that use agile approaches as well [Paulk 2014]. Hence, the needs of an organization has to be considered when using PRs to benefit from them correspondingly.

The assessment based on PRs also brings benefits for organizations. First, it increases their competitive strength on the IT market. Many organizations are faced with market pressures to respond multiple PRs [Garcia 2007]. Second, if multiple PRs are considered in parallel, then more points of references can be used for the software process improvement [Siviy et al. 2008a]. Consequently, the software process improvement is more effective. It is more efficient as well. The assessment based on PRs may not necessarily lead to more costs. The costs for such an assessment based on PRs are reduced as PRs are simultaneously considered [Ferreira et al. 2011; Siviy et al. 2008a].

1.3 Challenges with Multiple Practice Repositories

Multiple PRs may bring the mentioned benefits if they are properly used. However, three out of five organizations are facing challenges in using multiple PRs [Marino and Morley 2008]. There are two main challenges when working with multiple PRs [Kelemen 2013; Thiry et al. 2010]:

- Selection of the best PRs based on the internal needs and problems of organizations
- Simultaneous usage of multiple PRs to adopt and assess the adoption by the identification of similarities and dependencies of the selected PRs

An organization needs to select the PRs that can be relevant for its software process improvement.

As the term **PR** does not necessarily refer to all elements of a certain PR, the selection and the usage of PRs can refer to the selection of a subset of PRs' elements and the usage of these selected PRs' elements respectively.

Therefore, all or only some of their elements (e.g. all or only some of their processes) can be selected. For example, the continuous representation of CMMI proposes to select only the processes that are best suited for an organization.

There are *prescriptive* and *inductive approaches* to perform such a selection [Pettersson et al. 2008; Thomas and McGarry 1994].

According to the *prescriptive approaches*, the organizations select all elements of the considered PRs. For example, this approach is often applied in the automotive industry where certifications according to PRs are a selection criteria for suppliers. Unless imposed, we do not recommend its application as the adoption of all elements to get certified according to a PR often overwhelm the organizations and is associated with some risks. For example, the time and money of an organization are spent on the identification of a "flexible" assessor who is able to tolerate the weaknesses of the software processes in this organization [Mogilensky 2009]. Moreover, such organizations concentrate on the identification and adoption of the minimum acceptable elements to pass an assessment while the internal needs, problems and organizational culture are not properly considered [Mogilensky 2009]. These risks jeopardize the software process improvement effort and thus, the PRs do not deliver the desired benefits.

We recommend *inductive approaches* which are driven by the internal organization's needs and problems. No single PR is universally deployed or even universally useful [Jones 2007:13]. The PRs are too universal and comprehensive to address the local domains of organizations [Thomas and McGarry 1994]. Instead of selecting entire PRs, key concepts that address the specific needs and problems of organizations need to be selected [Coleman and O'Connor 2008; Heston and Phifer 2011]. Hence, the improved software processes are adjusted to the particular situation of an organization [Kautz 1998] to best fit its conditions and goals [Subramanian et al. 2009].

Once best suited PRs are selected for the software process improvement, an organization has to simultaneously adopt these PRs and perform assessments based on these PRs. As the PRs have similarities and dependencies, the organization has to manage their usage in a coordinated way and to benefit from synergy effects.

For an optimal usage of multiple PRs, their similarities have to be identified. Therefore, the redundancies between these PRs can be avoided in the adoption and assessment. Furthermore, the weaknesses of a single PR can be overcome by the strengths of other PRs. Consequently, the organizations can benefit from synergy effects for a better understanding and adoption of the PRs.

Furthermore, an optimal usage of multiple PRs is achieved if the dependencies within a PR or between multiple PRs are identified. Based on the time order between the PRs' elements, an organization can better manage the communication within and between different software areas.

In the following, we give more details about the selection of PRs and the identification of similarities and dependencies between PRs. We also mention some approaches that support these activities.

1.3.1 Selection Approaches

Based on a profound literature research regarding approaches to select PRs, we differentiate between *organization* and *project driven approaches*. Both are based on the needs and problems of organizations and thus, these are inductive approaches.

In an *organization driven approach*, the needs and problems are reflected by organization's goals, i.e. the business and IT goals. An organization whose primary goal is improving the time-to-market may follow a significantly different approach to software process improvement than one whose primary goal is to produce defect-free software [Thomas and McGarry 1994]. Consequently, the organization's goals are the input for the selection of best suited PRs.

The software process improvement alignment to the organization' goals is a critical success factor [Dyba 2005; Lepmets et al. 2012] and thus, various approaches are developed to address this challenge. Currently, two types of such approaches exist: *goal decomposition* and *goal characterization approaches*:

- *Goal decomposition* approaches propose to relate the goals at the last decomposition level to the PRs. One well-known approach that considers the goal decomposition is the GQM (Goal-Question-metric) [Basili 1992; Basili and Weiss 1984]. Here, the basic idea is to derive measures from measurement questions and goals and thus, perform a goals alignment with measures. The software process improvement alignment to the goals refers to the decomposition of goals until a relation with the PRs is possible. Some approaches use different methods for the goal decomposition. For example, the Function Analysis Systems Technique (FAST) with HOW/WHY-questions [Wixson 1999] and the Goal Categorization according to different decomposition levels [Basili et al. 2010] can be used for this purpose. Methods from requirements engineering domain to decompose high level requirements [Bresciani et al. 2004; Lamsweerde and Letier 2003] can be applied as well. Other approaches define concrete hierarchies of goals with a mapping to their corresponding PRs' elements. For example, IBM Measured Capability Improvement Framework [Trevellion 2009] or COBIT are such concrete approaches.
- *Goal characterization* approaches describe the goals of an organization using different factors or indicators to support the relation between these goals and PRs. For example, the CSFs approach uses critical success factors [Bullen and Rockart 1981], the MIGME-RCC approach uses business indicators to characterize the organizational goals and align strategies to these goals [Muñoz et al. 2013] and the Q-Genes define concrete capabilities of PRs to be mapped to the organization's needs [Heston and Phifer 2011].

According to the *project driven approach*, the software process improvement starts at the software project level. The software process improvement requires an intensive change management in an organization. Developers are usually motivated for change and thus, a bottom-up approach might better support the change management and the software process improvement initiatives [Conradi and Fuggetta 2002]. The project driven approach is based on the software projects' needs and problems which are reflected by their context.

Within an organization there are different types of software projects and software project contexts. There is no perfect PR for every type of organization. More, there is also not a single perfect PR for every type of software project. The reason for such a lack of universal utility of any single approach to software development is related to the basic requirement of a software process: it should fit the needs of the project [Feiler and Humphrey 1992]. For example, the agile or traditional PRs are not the solution for all types of software projects inside an organization. The agile PRs are mostly used for small, non-critical projects [Abrahamsson et al. 2009]. However, the agile practices may be

enriched with practices from traditional PRs according to the context in which they are used [Bowers et al. 2002; Cao et al. 2004; Lindvall et al. 2004; Stotts et al. 2003]. Analogously, the traditional PRs can be extended or tailored to fit a software project context.

Hence, we remind about the contingency theory that states that the optimal course of action has to be contingent with the needs and context where this course is applied. Consequently, the optimal software processes have to be contingent with the software project context. The established PRs also acknowledge that a software process should be designed to address the context in which the process operates. ISO/IEC 12207 recommends that the software processes should be appropriate to manage the different project constraints: project's scope, magnitude, complexity, changing needs and opportunities. CMMI-DEV adopts a similar position and recommends that various contexts should be considered when implementing processes. Hence, the role of the software project context have to be stressed for the process improvement to support the software projects in achieving their goals [Jeners et al. 2013b].

Based on a project driven approach, a software process improvement can be achieved at the project, as well as at the organizational level. At the project level, best suited PRs are selected according to the needs of a single project. The identification of best suited PRs motivates the software project members and leads to their adoption, and thus, to a process improvement. At the organizational level, best suited PRs can be selected according to the most critical needs of all software projects. This leads to a software process improvement that address the internal constraints of an organization and thus, leads to optimal software process that are contingent on the context [Benediktsson et al. 2006].

All types of PRs, the reference, process or internal process PRs can be considered for this selection. We remind that the term “PR” does not only refer to all the elements of a PR, but also to a subset of its elements. At the project level, the selected PRs’ elements can be directly adopted in the software project. At the organizational level, the selected elements from the internal process PRs can be analyzed for weaknesses and potential improvements if necessary. The selected elements from reference or process PRs can serve as guidance to define new elements in the internal process PRs or to improve the existing ones.

To summarize, we recommend an organization to select PRs based on the software projects’ needs and problems which are reflected by their context. In the absence of published guidance with respect to such complex decisions, we propose an approach for the alignment of the software project context to multiple PRs.

1.3.2 Identification of Similarities and Dependencies Approaches

Different PRs have similarities. One reason is that there are basic aspects that need to be considered by all types of software projects. For example, risk management has to be addressed by different types of software projects as their goals can be jeopardized by risks that are not handled. Some PRs describe these basic aspects in a more abstract manner, while others give more details about them (e.g. CMMI-DEV vs. V-Model XT). Another reason for the similarities between PRs is that some PRs are derived from other PRs. For example, SPICE is derived from PRs, such as ISO/IEC 12207 or SW-CMM [Paulk 1994]. More, an analysis of 52 PRs has shown that 73% of these PRs are defined based on existing PRs [von Wangenheim et al. 2010]. To identify the similarities and thus, the redundancies between PRs, these PRs need to be compared.

There exist various mapping materials where the similarities between PRs are documented, as well as mapping approaches that compare the PRs to identify their similarities and differences.

First, various mapping materials are provided to reveal the similarities and differences of the multiple PRs. For example, CMMI-DEV is compared with several PRs, such as ISO/IEC 12207, SPICE or with the agile PRs [Baldassarre et al. 2009; Ragaisis et al. 2010; SEIR 2013].

Second, there are systematic approaches to compare the PRs. These give guidelines on how to identify such similarities and differences. While some approaches propose to connect similar PRs' elements [Ferchichi and Bigand 2008; Malzahn 2009], other ones use metrics to compare the PRs. At a higher level, metrics such as the size and complexity are proposed to compare the PRs [Ferreira et al. 2010]. Here, the size is defined by the common process areas or differences in the description detail; the complexity is defined by the internal coupling and the dependencies between process areas. At a lower level, metrics such as the coverage degree and detail degree are proposed to compare the practices of PRs [Ekert et al. 2009]. The coverage defines the degree in which a practice covers another practice, the detail degree specifies which practice is more detailed and gives more information for its adoption.

There are also dependencies within a single PR or between different PRs. These have to be identified. Only some PRs give information about these dependencies. For example, there exist learning materials for CMMI-DEV, where the dependencies between its process areas and practices are given. There are also some systematic approaches that support the identification of dependencies between practices. These are based on the inputs and outputs of the practices [Chen et al. 2008].

1.4 MOSAIC Approach

Activities, such as the selection of PRs, the identification of similarities and dependencies between them support organizations in their software process improvement. We developed the MOSAIC approach (in short MOSAIC) to support organizations to perform these activities and achieve an effective and efficient adoption of PRs and assessment based on PRs.

MOSAIC is based on the project driven approach. As the name suggests, MOSAIC integrates various puzzle pieces – the multiple PRs and the software project context. It is a systematic and extensible tool supported that aims to achieve the following goals:

- (G1) Automated selection of practices from multiple PRs based on the software project context
- (G2) Automated identification of two or more similar practices from multiple PRs
- (G3) Automated identification of practice dependencies of multiple PRs

These goals address the two main challenges for organizations that work with multiple PRs:

- Selection of the best PRs based on the internal needs and problems of organizations
- Simultaneous usage of multiple PRs to adopt and assess the adoption by the identification of similarities and dependencies of the selected PRs

Within MOSAIC, practices are the elements that can be selected and compared. First, the practices are the basis for concrete activities that can be selected and performed within the software process improvement. Second, the PRs are comparable on the practice level as the PRs have high similarities between their practices. Furthermore, this comparison at the practice level allows to identify the practices dependencies as one practice defines an output that is requested as input for another practice.

We aimed to offer various automations of the afore mentioned activities to increase the time efficiency of organizations working with multiple PRs. Consequently, we implemented the MOSAIC Toolbox to prove that such automations are possible.


Remark

Within MOSAIC, we focused on the software development, but aimed to develop a flexible approach that allows the integration of PRs for other software areas.

MOSAIC is not limited to PRs for software development. It also aims to consider PRs for other software areas due to the dependencies and similarities between them.

First, the software development is strongly related to the other software areas and thus, there are dependencies between the PRs for the different software areas. The lifecycle of a software product does not contain only the software development phase. For example, after the initial development of the software product, the software operation (or service operation according to ITIL) is performed. The identification of dependencies between the PRs can be used to manage the dependencies between the departments of an organization responsible for the different phases in the software lifecycle. As a concrete example, not all software errors that are discovered during the verification of a software are usually resolved during the software development. The software operation needs to have as much information as possible about these errors in order to resolve them. Consequently, a strong dependency related to the software errors exists between the software development and operation. Hence, organizations that produce software products should not consider only PRs for the software development, but also PRs for other software areas.

Second, organizations should consider PRs for different areas to improve all their software processes in a coordinated way. Although the PRs are defined for different software areas, they address similar aspects. Therefore, there exist mappings between the PRs for different software areas. For example, institutions that define PRs, such as the Information Systems Audit and Control Association (ISACA) or the Software Engineering Institute (SEI) offers mappings between COBIT and CMMI-DEV or between CMMI-DEV and ITIL [ISACA 2011b; SEIR 2013].

Hence, we developed an approach that is valid for the software development, but that can consider PRs for other software areas too.

1.5 MOSAIC Challenges, Research Questions and Working Fields

There are different challenges that need to be addressed to achieve the MOSAIC goals. Inspired by the Goal-Question-Metric (GQM) approach [Basili and Rombach 1988], we derive research questions and sub-questions (marked with RQ) to address the challenges for each goal. The answers to these questions are reflected by working fields (WF) that need to be addressed to achieve the goals.

The *automated selection of practices* from multiple PRs based on the software project context requires an integration of the multiple PRs with this context. First of all, a characterization of the software project context is needed to connect it to the multiple PRs. As the software project context differs from one software project to another, an analysis of frameworks that describe this context

needs to be performed. Another challenge is the development of a systematic approach for an automated selection to be implemented in the tool support. Hence, the research questions and working fields for the first goal (G1) are:

- (RQ1.1): h
  - (WF1.1.1): Analyze various frameworks that describe the software project context.
  - (WF1.1.2): Select a framework that characterizes the software project context.
- (RQ1.2): How can the software project context be integrated with the multiple PRs?
  - (WF1.2): Develop an approach to connect the selected framework that characterizes the software project context to practices of considered PRs.
- (RQ1.3): How can the selection of practices be automatically performed?
  - (WF1.3): Develop an approach to systematically and automatically select practices based on the software project context.

Second, the *automated identification of similar practices* and the *identification of dependencies between them* require the integration of multiple PRs. This is a challenge. According to ISO/IEC 24744 [ISO/IEC TR 24774:2010 2010], the PRs vary in format, content and level of prescription and thus, have different structure and terminology [Andelfinger et al. 2006; Siviy et al. 2008c]:

- *PRs are organized using different structures*. The International Organization for Standardization (ISO) tries to promote the definition of PRs with a common structure. Some of PRs, such as ISO 9001 and ISO 90003 have a similar structure [Kelemen 2013]. SEI also tries with the CMMI constellation to achieve a common structure for the PRs. However, we can argue that most of the PRs are still organized using different structures. Often PRs are using different names for the same structural element. For example, the CMMI-DEV practices are called specific or generic practices while in ISO/IEC 61508 they are called requirements. While a group of processes addressing the same topic is called domain in COBIT, in CMMI-DEV it is called a category. Furthermore, PRs are written on different levels of abstraction. We found reasonable similarities between COBIT control objectives, COBIT control practices, CMMI specific goals, generic goals, practices, sub-practices, SPICE practices and IESO/IEC 61508 objectives and requirements. Without information about this structural mapping and thus, a normalization of the structure, it is not possible to automatically identify similar practices.
- *PRs are described in different terminology*. PRs use different terms to express the same semantic concept. For example, the terms “noncompliance issues” in CMMI-DEV and “non-conformances” in SPICE or “risk associated with project life cycle” in CMMI-DEV and “project risks” in SPICE refer to the same concept. In addition, each PR uses its specific writing style. In some of the PRs the verbs are used in their active form, while other PRs make extensive use of passive, gerunds or nominalizations. This hampers the identification of different terms that are semantically similar and does not allow an automated identification of similar practices.

Besides the different structures and terminology, each PR contains a considerable amount of data. Consequently, an integration of several PRs that allows automated operations becomes more complex. Therefore, a suitable tool support is needed to assist the integration of this data.

Furthermore, the integration of multiple PRs must be flexible so that PRs for different software areas can be integrated.

Hence, we define the research questions and working fields related to the integration of multiple PRs that are necessary to achieve the second and third goal (G2 and G3):

- (RQ2-3.1): How can the multiple PRs be integrated?

- (WF2-3.1.1): Analyze various related approaches to integrate the multiple PRs and allow the identification of similar practices and of dependencies.
- (WF2-3.1.2): Provide a flexible design to integrate PRs for other software areas.
- (WF2-3.1.3): Analyze and offer different tool support to integrate the significant amount of data.

Based on this integration, systematic approaches are possible to automatically identify similar practices and dependencies between them.

The development of an automated systematic approach to identify similar practices is also a challenge. First, the comparison of not only two, but of more practices needs to be particularly investigated. Furthermore, to define when practices are similar, an analysis of the similarity between objects in general is needed. Finally, a tool supported approach has to be developed to allow the comparison of two or more practices. Therefore, we define the next research questions and working fields for the second goal (G2):

- (RQ2.2): How can similar practices be automatically identified?
  - (WF2.2.1): Analyze similarity theory to identify when objects are similar and how can we identify their similarity.
  - (WF2.2.2): Identify different modalities to compare two or more practices and identify their similarity.
  - (WF2.2.3): Develop an approach to systematically and automatically identify similar practices.

The development of an automated systematic approach to identify the practice dependencies is also needed as many PRs do not explicitly define these dependencies within a PR or between PRs. Analogously to the identification of the similar practices, we define the next research questions and working fields for the third goal (G3):

- (RQ3.2): How can dependencies between the practices be automatically identified?
  - (WF3.2.1): Develop an approach to connect practices that are dependent.
  - (WF3.2.2): Develop an approach to systematically and automatically identify the dependencies between practices.

1.6 Thesis Structure

In the following, we give a short description of the next chapters.

While the **chapter 1** motivates the usage of multiple PRs for the software process improvement initiatives and introduces MOSAIC and its goals this usage, the **chapter 2** presents an overview of MOSAIC by describing its parts and organizational roles who can use MOSAIC. Before introducing the MOSAIC parts and the involved roles, we give an example scenario where an organization is interested to use certain PRs (section 2.2). We use the practices from these PRs to motivate MOSAIC (section 2.3) and later in the next chapters to illustrate its development (chapters 4, 5 and 6).

We continue with a short description of the MOSAIC parts. In the section 2.3, we give a short overview of the analysis activities with their corresponding metrics. We also give examples of practices from our example scenario to motivate these analysis activities and to support their understanding. In section 2.4, we describe the models and modeling activities. This description also contains a review of the most important related work. This review is performed to differentiate between the

existing approaches and the contribution of MOSAIC. In section 2.5, an overview of the implementation of MOSAIC, the MOSAIC Toolbox, that implement the modeling and analysis activities, is given.

We close with a description of the roles that can use MOSAIC and define a mapping between roles in organizations and the MOSAIC roles (section 2.6).

In the next chapters (chapters 3, 4, 5 and 6), we describe in more detail the MOSAIC parts.

The **chapter 3** describes the MOSAIC meta-models with its elements and relations between them. MOSAIC consists of three meta-models. While the first meta-model is defined to normalize the structure of various PRs (section 3.1), the second one is defined to normalize the terminology of these PRs (section 3.2). The third meta-model is used to integrate the software project context with the PRs (section 3.3).

The **chapters 4** and **5** describe the modeling and the analysis activities respectively.

In the chapter 4, we describe the modeling activities that can be performed to create the MOSAIC models. The MOSAIC models are created according to the afore-mentioned meta-models. Consequently, we describe modeling activities to normalize the structure of the various PRs (section 4.2), to normalize their terminology (section 4.3) and finally to integrate the software project context with these PRs (section 4.4). In the description of the modeling activities, we also define several guidelines to create these models. These guidelines are based on our experience with the integration of multiple PRs. We exemplify the modeling activities by modeling some practices from our example scenario (sections 4.1).

In the chapter 5, we describe different analysis activities that are based on metrics. After a short introduction of the measurement theory (section 5.2), we describe the purpose of the metrics and a design principle that is valid for all these metrics (section 5.3). In the next sections, we define for each analysis activity the proposed metrics and how these metrics are used. We describe the selection of practices based on the software project context (section 5.4), different modalities for the identification of similar practices (section 5.5) and the identification of dependencies between practices (section 5.6).

The **chapter 6** lists numerous activities where MOSAIC can be applied to support organizations to work with one or more PRs. For each such activity, we describe the MOSAIC application and its limitations.

The **chapter 7** gives details about the MOSAIC Toolbox. We describe the requirements for the MOSAIC Toolbox (section 6.1). Then, an overview of the different tools within the MOSAIC Toolbox is given (section 6.2). These tools are integrated into a web application whose architecture is described in section 6.3. Finally, section 6.4 gives some details about the implementation of this architecture. More details about the MOSAIC Toolbox (e.g. how the different tools can be used) can be found in the appendices of this work.

The **chapter 8** presents the evaluation activities performed to verify the achievement of our goals. Experiments (section 8.3), case and field studies (section 8.4) that involve different experts from industry and research are described. The threats to validity, the limitations of MOSAIC and the future work are also mentioned. Furthermore, we present the evaluation of the MOSAIC Toolbox according to its quality in use, i.e. evaluation of its effectiveness (section 8.5.1), efficiency (section 8.5.2), satisfaction and usability (section 8.5.3). Based on this evaluation, we give some examples of the future work (section 8.5.4).

The **chapter 9** closes this work with the MOSAIC contributions to the software community (section 9.1) and with a summary of this work (section 9.2).

2 Overview

In the following, we present an overview of MOSAIC and exemplify how it can be used by organizations. In the description of MOSAIC, we make references to the related work and describe the differences between the existing approaches and the contribution of MOSAIC. We close with a short description of the MOSAIC Toolbox.

2.1 Approach

MOSAIC consists of **models** (incl. their meta-models), **metrics**, **tool supported activities** and offers support for different **roles** that have to deal with PRs (Fig. 1).

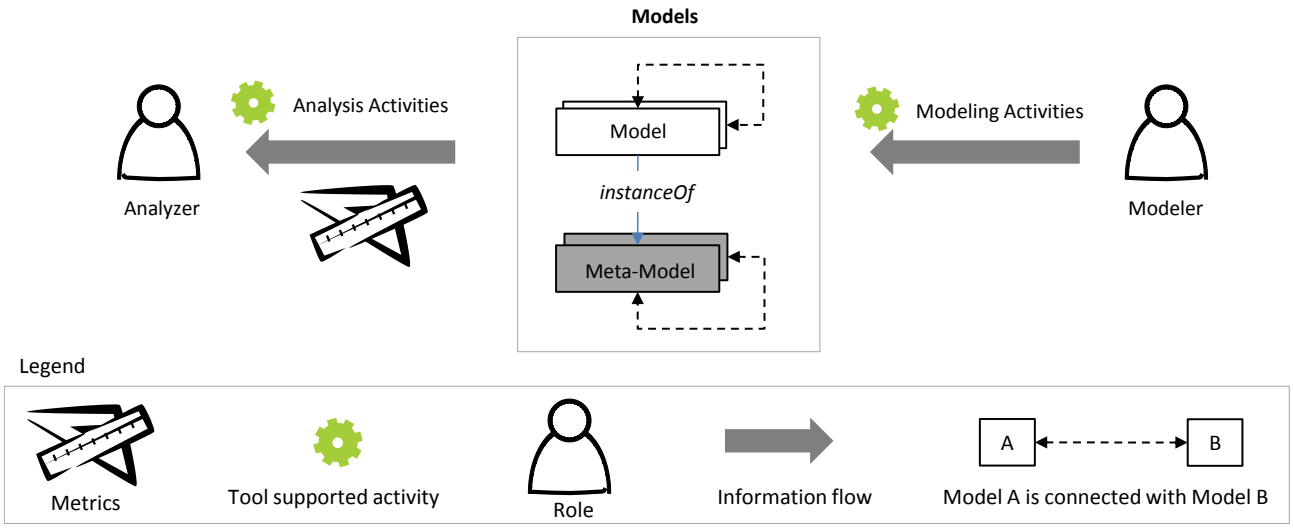


Fig. 1: MOSAIC parts

The models represent the multiple PRs and the software project context and are connected to integrate these PRs with each other and the PRs with the software project context. Section 7.6.1 gives an overview of commonly used PRs that could be integrated into MOSAIC.

The models are created and then used by two roles respectively. These roles can perform tool supported activities. The **Modeler** performs the **modeling activities** to create the models. The **Analyzer** uses these models and performs the **analysis activities** to make decisions regarding the adoption of PRs and the assessment based on PRs. He selects practices based on the software project context, identifies similar practices of PRs and dependencies between them.

We defined systematic approaches for each of the afore-mentioned analysis activities. Therefore, we defined metrics to obtain differentiable, reproducible and comparable results for each such analysis. For example, we defined similarity metrics for the identification of similar practices. These metrics can be integrated in various algorithms to implement an analysis activity. In this work, we give examples of such algorithms. Such an example is the computation of the similarity degree between two or more practices based on the similarity metrics. An example that we did not implement is the identification of a mapping between the processes of two PRs. Such implementation is possible in MOSAIC based on these similarity metrics.

As the models are an important part of MOSAIC, we give an overview of the MOSAIC way to integrate PRs and the software project context (Fig. 2). This is based on an idea developed during the first stages of this work [Jeners (Pricope) and Lichter 2011], evaluated later during a master thesis of a student at our research department [Gómez Rosenkranz 2010] and published [Jeners et al. 2013c].

We defined three meta-models, the **Integrated Structure Meta-Model** (IS Meta-Model), **Integrated Concept Meta-Model** (IC Meta-Model) and **Situational Factors Meta-Model** (SF Meta-Model).

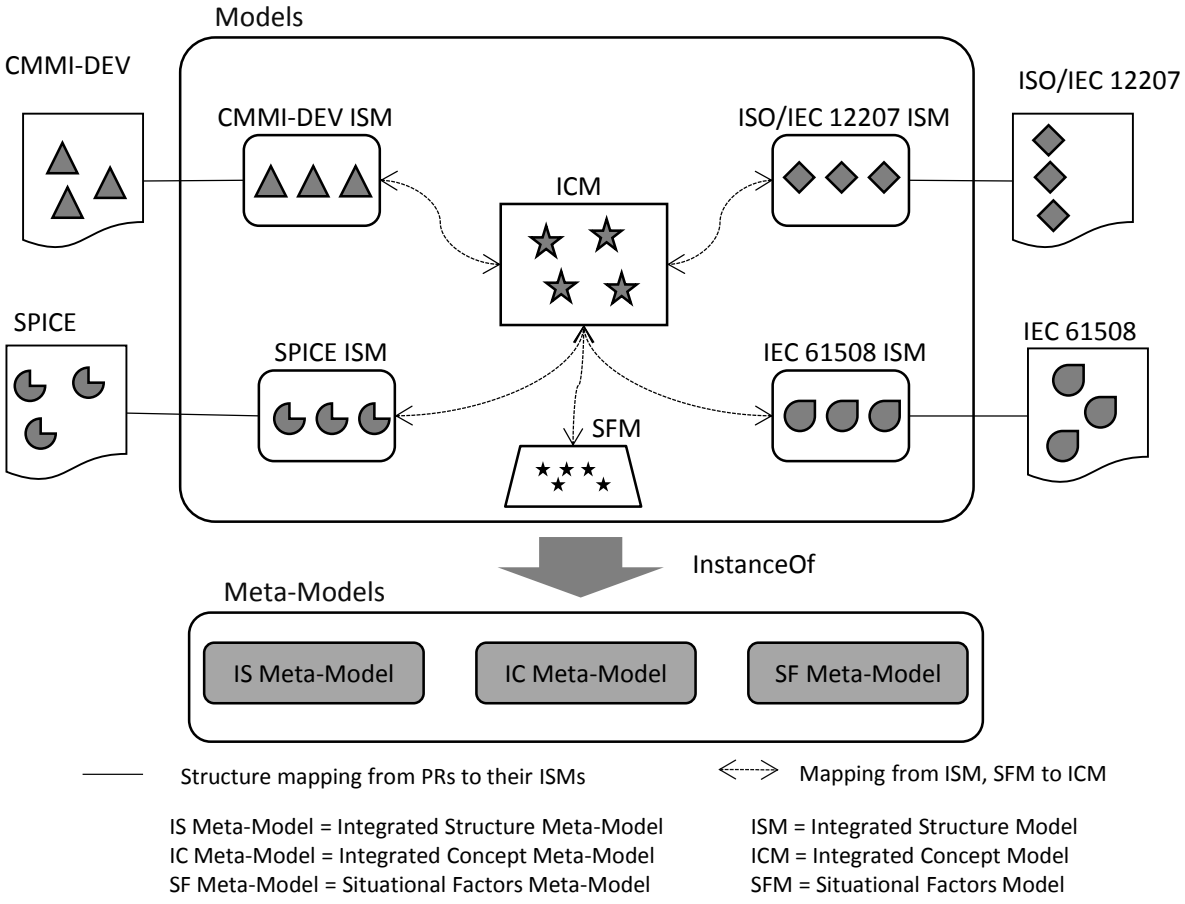


Fig. 2: MOSAIC models – An overview

First, the IS and IC Meta-Models are used by a Modeler to integrate the structure and the terminology of different PRs. Fig. 2 depicts the purpose of both meta-models and their respective concrete models, the ISMs and ICM respectively. The illustrated ISMs are only examples for PRs that can be modeled in MOSAIC.

The PRs' elements are represented by small geometrical shapes inside each PR. While the different positions of these shapes inside a PR symbolize the different structures of PRs, their different forms symbolize the different terminology of these PRs. For each PR, a Modeler normalizes its structure and creates its corresponding ISM where the shapes have the same position and thus, the shape of the ISMs have the same structure. Hence, all ISMs use the same structure which makes them comparable. Furthermore, the Modeler normalizes the terminology of the different ISMs and create a common ICM where the shapes have the same form. These shapes of the same form are connected

with the shapes of different forms in the ISMs if different terms have the same meaning. Hence, the ISMs use the same terminology which makes them automatically comparable.

Furthermore, the SF Meta-Model is used to characterize the software project context and to integrate it with the PRs. For this purpose, the Modeler connects the SFM elements to the ICM elements and thus, to the ISM elements for each PR. Consequently, the ISMs and the SFM are connected and this also makes them automatically analyzable.

To summarize, ICM is a central model that supports the integration of the PRs with each other and with the software project context by connecting their elements. Hence, it is the most important model of MOSAIC.

The final ideas and results of MOSAIC were presented in a journal paper [Jeners et al. 2013b] and at a conference talk [Jeners 2014]. The journal paper is resulted based on a collaboration with four academic researchers after a presentation and intensive discussion about MOSAIC.

2.2 Example Scenario

In this section, we give an example of an organization that is interested in the usage of multiple PRs. For a better understanding of MOSAIC motivation and its support for organizations, we describe a possible scenario where MOSAIC can be applied.

The application quality is the most important goal of a medium size organization with 200 employees. It produces software devices to be integrated in automobiles. Consequently, the quality is of major concern for the software projects within this organization. However, a high percentage of its software projects do not achieve the requested quality. The results of an external assessment indicate poor software processes as a reason for this problem. Currently, the organization has a more general internal process PR to serve as a rough guideline for the software development in the projects. But it does not have any experience with reference or process PRs.

Based on the recommendations of the external assessors and on the process profiles of their customers, the organization decides to start a software process improvement initiative according to CMMI-DEV, SPICE, ISO/IEC 61508 and ITIL. While some of its customers use CMMI-DEV, others use SPICE for their software development. The usage of the same PRs as its customers strengthens the relations between these customers and this organization. For example, the communication between them becomes standardized and more efficient. Furthermore, the external assessors recommend to use ISO/IEC 61508 as the functional safety of the software devices is an important requirement in the automotive industry. Finally, the external assessors recommend to consider ITIL as the software operation processes are strongly related to the software development. More, the software operation processes might support to achieve the requested quality as well.

As there are no requirements from its customers to use any PR, the organization decides to select only parts of the afore-mentioned PRs for their software process improvement initiative. Therefore, the challenge now is to select practices from these PRs that are best suited to support the software projects in managing the requested quality. Furthermore, the considered PRs have similarities and dependencies. Consequently, the organization needs to identify them to manage the usage of the PRs in a coordinated way and benefit from synergy effects. For example, redundancies have to be avoided, information about the adoption of selected practices from the multiple PRs is needed, the time order between the practices needs to be defined, as well as the information flow between the software development and operation has to be known and specified.

In the following, we use practices that are relevant for this organization to motivate MOSAIC and illustrate how it can be used. We do not aim to select all practices from the considered PRs or

show all their similarities and dependencies, but only give some examples for a better understanding of MOSAIC. In this chapter, we motivate MOSAIC by giving examples of how the analysis activities are applied. In the next chapters, we illustrate how the MOSAIC models related to some practices are created, how the selection, dependencies and similarities identification of these practices are implemented, as well as how these modeling and analysis activities are performed using the MOSAIC Toolbox.

2.3 Analysis Activities and Metrics

In this section, we describe the MOSAIC analysis activities, their related metrics, as well as examples of practices to illustrate these activities and thus, how MOSAIC can support Analyzers and their organizations in the software process improvement.

2.3.1 Selection of Practices

There is a lack of published guidance related to the selection of best suited practices from multiple PRs. Therefore, we defined an approach to automatically select practices from multiple PRs based on the software project context. For this purpose, we defined metrics that determine the support degree of a practice for a software project context. We differentiated between a “Strong”, “Medium” and “Absent” support degree of a given practice for a certain software context. Practices with a “Strong” or “Medium” support degree can be selected to be adopted. These can be applied by Analyzers in the software projects or can be considered for the improvement of the internal process PRs. At last, these practices will help software projects to manage current or possible critical situations that exist or can appear during project phases.

Hence, MOSAIC supports the selection of such practices. For a certain software project context, MOSAIC delivers as result the best suited practices and their corresponding support degree for this project context.

2.3.1.1 Examples

The organization from our example scenario is interested in the usage of CMMI-DEV, SPICE, ISO/IEC 61508 and ITIL. As there are no requirements to apply the entire PRs, only practices that are best suited to address the application quality are of interest.

Table 1 lists some examples of practices from CMMI-DEV, SPICE, ISO/IEC 61508 and ITIL that can be selected with MOSAIC to support the software projects to manage the application quality. We do not aim to give all practices from the considered PRs that are related to the application quality, but only some to exemplify and motivate the MOSAIC results.

The involvement of eligible stakeholders in the resolution of quality issues strongly supports a software project to achieve a high application quality (example 1). The managers have the appropriate authority to support the staff to solve these quality issues. These quality issues do not only refer to product issues but also to process issues. Poor process quality leads to a poor application quality and thus, these also need to be intensively considered.

Another example of a strong support to achieve a high application quality might be a systematic selection of a strategy for the verification of the software (example 2). Best suited techniques, e.g. static/dynamic analysis, code inspection/review, white/black box testing, code coverage have to be

systematically chosen to verify the software units and thus, identify the software faults that need to be resolved.

Example 3 indicates a concrete verification technique, namely the boundary value analysis technique. The search of software errors occurring at parameter limits or boundaries is a well-known verification method to detect software errors.

An analysis of requirements about the needs and constraints of the stakeholders is also recommended (example 4). This analysis does not strongly support a software project but it should be adopted as the requested quality can influence the involvement of the stakeholders. If the stakeholder needs and constraints are not balanced due the quality constraint, conflicts can appear in the project. Such conflicts might jeopardize the achievement of the project goals and thus, the project have to pay attention about the impact of the requested quality.

Finally, example 5 illustrates an ITIL practice that also plays an important role for a high application quality. One of the usages of the CMS (Configuration Management System) is the documentation and providing of information about incidents and problems, such as faulty equipment, impact of incidents and problems or information about the responsibilities for solving these incidents and problems. Therefore, the CMS provides the relevant information to find a solution for failures and thus, to achieve the requested quality.

Situation in organizations or in a software project	Example	Selected practices	Support Degree
high application quality	1	**CMMI-DEV PPQA SP2.1** Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers.	Strong
	2	**SPICE ENG.6.BP1** Define a unit verification strategy.	Strong
	3	**ISO/IEC 61508 Part 3 C.5.4** Use Boundary value analysis.	Strong
	4	**CMMI-DEV SP3.4** Analyze requirements to balance stakeholder needs and constraints.	Medium
	5	**ITIL Service Operation 4.4.5.5** The CMS (Configuration Management System) must be used to help determine the level of impact and to assist in pinpointing and diagnosing the exact point of failure (..).	Strong

Table 1: Example – Selected practices to manage the application quality

2.3.2 Identification of Similar Practices

Considerable research has already been performed to compare the multiple PRs. However, the existing approaches are not objective enough or the reasoning for similarities between practices is missing. Therefore, the results are not repeatable. Furthermore, the comparison is not detailed enough to understand the practice similarities. Finally, approaches are missing for an automated comparison of practices.

We performed an analysis of the similarity theory and of the existing methods to develop an approach to identify when practices are similar. In general, objects are similar, if their features are similar and thus, practices are similar if their elements are similar. We also identified some methods from the similarity theory to compute the similarity between its practice elements. Finally, we analyzed which information is relevant for organization to offer several modalities to compare and identify similar practices. These are only examples, as other algorithms to identify similar practices are possible as well. We defined the following algorithms to compare two or more practices:

- Identification of the similarity between two or more practices
- Identification of the coverage of two sets of practices
    - Identification of the highest coverage in a set of practices
    - Identification of the best coverage in a set of practices
- Identification of the output states of practices

The *similarity between two or more practices* is defined by the similarity degree between these practices. Therefore, we defined metrics that determine this similarity degree. This degree defines how many similar elements the practices have in common. We differentiated between "Equal", "High", "Medium", "Low" or "Non-Equal" similarity degrees. While the identification of practices with a "High" or "Equal" similarity degree mainly support the organizations to avoid the redundancies, the practices with a "Medium" or "Low" degree support them to benefit from synergy effects. Synergy effects are given by the differences between these practices with a "Medium" or "Low" similarity degree as these differences can serve as additional useful information about how to adopt them.

The *coverage between two sets of practices* is defined by the coverage degree of the first set of practices in the second set of practices. Therefore, we defined metrics that determine this coverage degree. This degree defines how many elements of the first set of practices cover the elements of the second set of practices. The coverage degree has a value between 0 and 1. It does not only indicate that practices are similar, but also that they cover each other. The difference between the coverage and similarity degree is that if the coverage between two sets of practices is high, it does not have to be that the similarity between these practices is also high. One set can contain the elements of the other set and also other elements. Then, their coverage degree is 1, but their similarity degree can be "Low", "Medium" or "High", but never "Equal". If a set of practices covers another set of practices, redundancies can be avoided in the adoption and assessment. This is because, only the adoption of the first set of practices and the assessment according to these practices are needed. Furthermore, based on these coverage degrees between sets, an Analyzer can also identify the practice with the *highest* and the subset of practices with the *best coverage in one set of practices*.

- A practice with the highest coverage has the maximum coverage degree in a considered set of practices. When an organization needs to adopt or assess more practices, it can choose the practice with the highest coverage degree to cover as much as possible from the entire set of practices.
- A subset of practices with the best coverage refers to the minimum number of practices with a coverage degree of 1 in a considered set of practices. Therefore, an organization can only concentrate on these practices in the adoption and assessment to cover the entire set of practices.

The *output states of practices* is defined by the state of their outputs. Practices that produce similar outputs are similar. We defined metrics that determine this output state based on the Deming-Cycle values: Plan-Do-Check-Act. The Deming-Cycle is an iterative four-steps improvement process, that starts with the planning of a process (Plan), continues with its implementation (Do) and its verification against expected results (Check). By deviations, the root causes are identified so that corrective actions can be derived and applied in the next cycle (Act). Inspired by this idea, we differentiated between "Plan", "Do", "Check" or "Do-Check" output states to offer information about the lifecycle of a certain output produced by a practice. These states reflect whether the output is created ("Plan"), implemented ("Do"), verified ("Check") or verified and updated ("Do-Check"). Therefore, organizations can benefit from synergy effects as they can use information from different PRs to adopt these outputs.

Hence, MOSAIC supports the identification of such similarities by indicating the *similarity degree*, the *coverage degree* and the *output states between practices*.

2.3.2.1 Examples

The organization in our example scenario need to use CMMI-DEV, SPICE, ISO/IEC 61508 and ITIL and to identify their similar practices that are related to the application quality. The identification of the similarities and differences of these practices help the organization to manage their adoption in a coordinated way and to benefit from synergy effects. We give some examples of practices that have different similarity degrees, cover each other or are similar concerning a certain output. Analogously to the selection of practices, we do not aim to identify all similarities of the considered PRs, but only to show how redundancies can be avoided or how additional information for the adoption of practices can be gained.

Table 2 presents some examples of the *similarity between two or more practices* of the PRs considered by the organization.

Example 1 and 2 illustrate practices with “Equal” or “High” similarity degrees that need to be identified to avoid the redundancies between the multiple PRs. Example 1 lists “Equal” practices related to the tracking and documentation of quality assurance activities. Example 2 illustrates “High” practices that address the resolution of process issues. These practices also differ. The CMMI-DEV practice specifies that the quality issues have to be communicated to and solved together with various stakeholders. The SPICE practice does not give such information.

Example 3 and 4 illustrate practices with “Medium” and “Low” similarity degrees. These have differences that can serve as additional information for the adoption. In example 3, the SPICE practice gives concrete details about how to resolve issues. In example 4, while the SPICE practice requires that errors are documented, the ITIL practice gives concrete details about their documentation. It requires that the errors that cannot be solved during the software development and are accepted for release have to be properly documented for their resolution. All these differences lead to a “Low” similarity degree. Although ITIL is mainly used for the software operation, it can also provide helpful information for the software development.

Example 5 illustrates three practices with a “High” similarity degree that can be used to avoid the redundancies between the multiple PRs. MOSAIC can identify the similarities of more than two practices. For this comparison, MOSAIC does not consider each two pairs of practices, but all the practices at once. This is possible as MOSAIC identifies the similar elements contained in all these practices. Consequently, the similarity degree of all practices is calculated based on the similarity degree of their elements. In the example 5, the documentation of non-conformances is required by all three practices. Therefore, their similarity is “High”.

Example	Practices	Similarity degree	Similarities	Differences
1	**CMMI-DEV PPQA SP2.2** Establish and maintain records of quality assurance activities. **SPICE SUP.1.BP7** Track and record quality assurance activities.	Equal	Tracking and documentation of quality assurance activities	-
2	**CMMI-DEV PPQA SP2.1** Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers. **SPICE SUP.1.BP9** Ensure resolution on non-conformances.	High	Resolution of issues	- Communication of overall quality issues - Resolution of process with certain stakeholders
3	**CMMI-DEV PPQA SP2.1.1** Resolve each noncompliance with the appropriate members of the staff if possible **SPICE SUP.1.BP10** Implement an escalation mechanism.	Medium	Resolution of issues	Concrete resolution of issues by escalation mechanisms
4	**SPICE SUP.2.BP3.2** The results of verification activities are recorded. **ITIL Service Operation 4.4.5.11** Where a decision is made to release something into the production environment that includes known deficiencies, these should be logged as Known Errors in the KEDB, together with details of workarounds or resolution activities.	Low	Documentation of errors after the verification activities	- Analysis of errors to decide which are not resolved (Known Errors) - Documentation of Known Errors for the software operation - Identification and documentation of workarouns and resolution activities
5	**CMMI-DEV PPQA SP2.2** Establish and maintain records of quality assurance activities. **SPICE SUP.1.BP7** Track and record quality assurance activities. **ISO/IEC 61508 Part 3 7.9.2.5** After each verification, the verification report should address (..) non-conformances.	High	Documentation of non-conformances	Tracking of the non-conformances

Table 2: Example – Similarity between practices

Table 3 illustrates the *coverage between practices*.

In the example 1 and 2, only one practice from the two practices entirely covers the other one. The coverage degree of the CMMI-DEV practice is 1, i.e. the CMMI-DEV covers the SPICE practice. Therefore, the adoption of the SPICE practice and eventually the assessment of the internal process PRs according to this practice is not necessary if the CMMI-DEV practice is considered. The SPICE practice does not cover the CMMI-DEV practice. Therefore, the CMMI-DEV has the highest coverage degree in the set of the CMMI-DEV and SPICE practices.

Example 3 illustrates that one practice can cover more practices. The CMMI-DEV practice covers the two SPICE practices. Therefore, the best coverage is given by the CMMI-DEV practice in the set composed of the CMMI-DEV and the two SPICE practices. This is because, one practice covers all three practices. Therefore, an organization can only concentrate on the CMMI-DEV in the adoption and assessment.

Example	Practice	Coverage degree	Practice	Description
1	**CMMI-DEV PPQA SP2.1** Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers.	1.00	**SPICE SUP.1.BP9** Ensure resolution on non-conformances.	The CMMI-DEV practice covers the SPICE practice.
2	**SPICE SUP.1.BP9** Ensure resolution on non-conformances.	0.64	**CMMI-DEV PPQA SP2.1** Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers.	The SPICE practice does not entirely cover the CMMI-DEV practice.
3	**CMMI-DEV CM SP2.1** Track change requests	1.00	**SPICE SUP.10.BP5** Establish the dependencies and relationships to other change requests. **SPICE SUP.10.BP6** Assess the impact of the change.	The CMMI-DEV practice covers the SPICE practices.

Table 3: Example – Coverage between practices

Table 4 illustrates the *output states of practices*. Practices with "Plan", "Do" and "Check" output states are given. Therefore, different information about the adoption of this output is provided. The "quality issues" need to be documented in the verification report ("Plan"), to be analyzed ("Check") and resolved ("Do").

Practice	Output State
ISO/IEC 61508 Part3 7.9.2.5c After each verification, the verification report adresses non-conformances.	Plan
SPICE SUP.2.BP4 Determine and track actions for verification results.	Plan, Check
CMMI-DEV PPQA SP2.1 Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers.	Do

Table 4: Example – Similar practices – Output states

2.3.3 Identification of Practice Dependencies

We developed an approach to automatically identify the dependencies between practices of multiple PRs. This approach considers the inputs and outputs of practices to identify their dependencies. If a practice needs as input the output produced by another practice, then the practices have a dependency. We defined metrics that determine the dependency degree between two practices. We differentiated between "Strong", "Medium" or "Absent" dependency degrees to indicate how dependent are the practices between each other. These dependency degrees have to be considered to manage the dependencies between practices within the software development area and between the software development area and other software areas.

Hence, MOSAIC supports the identification of dependencies between practices. For a certain practice, MOSAIC delivers as result its dependent practices from PRs of interest with a "Strong" or "Medium" dependency degree.

2.3.3.1 Examples

The organization in our example scenario needs to identify the dependencies between the practices of the considered PRs to manage the usage of the PRs in a coordinated way. Fig. 3 illustrates some examples of dependencies between practices that can help organizations to manage the information flow between practices that are selected from a PR or from different PRs.

Example 1 and 2 illustrate dependencies between practices of a process and of different processes of a PR. In example 1, the noncompliance issues must be first identified and then communicated. Therefore, the time order for the adoption is defined. Moreover, this information can also be used to efficiently perform assessments. For example, the assessment according to the CMMI-DEV PPQA 2.1 practice is not necessary, if CMMI-DEV PPQA SP1.5 practice is not properly adopted. The dependency degree between these practices is "Strong" as their input and output are semantically equal. In example 2, quality issues are identified during peer reviews and then are communicated. Analogously, the time order is identified. The dependency degree is "Medium" as their input and output are similar but not equal. The "review issues" identified during the verification of the software products are a special case of "quality issues". There can be other types of quality issues, such as issues related to process non-conformances. For example, there is an issue if the checklists that are used for reviews do not contain enough criteria to verify a requirements specification.

Example 3 illustrates dependencies between PRs for different software areas. Analogously to the identification of similar practices, the PRs for other software areas should be considered as the software development is strongly related to them. We consider an example of dependencies between CMMI-DEV and ITIL. We assume that an error is not solved during the software development. Its resolution during the software operation can be efficiently performed if during the software development enough information is provided. Information about possible error causes, verification criteria or environment is necessary for the software operation to efficiently close the problem.

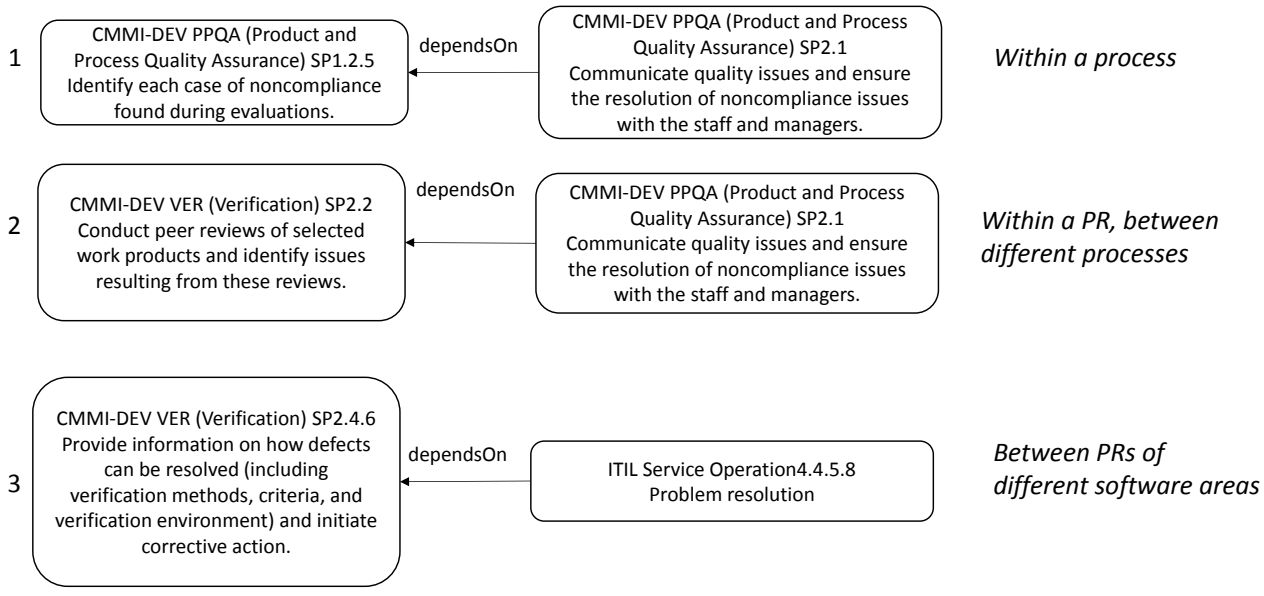


Fig. 3: Example – Practice dependencies

2.4 Models and Modeling Activities

As the name suggests, MOSAIC integrates various puzzle pieces – the multiple PRs and the software project context. MOSAIC is a model based approach that supports this integration by modeling and connecting the PRs and the software project context.

Models are a restricted representation of the reality [Ludewig 2002], and are a functional, structural and behavioral analogy of the original. They are created as it is not possible or it is too costly to build the original exactly how it appears in the reality [Alisch 2005].

As there is no standardized model for the software project context, such a model has to be created. The PRs are models by their definition. However, these models are organized using different structures. Consequently, new models of the PRs must be created as well. For this purpose, we defined their meta-models. According to the mega-modeling theory [Favre 2005], a meta-model is a "model of a model". The fact that a model is a meta-model is not an absolute characteristic of the model itself, but the role played by the model with respect to other models. There is an "instanceOf" relation between the models and their meta-model. Meta-models define the syntax and semantics of a modeling language [Pohl 2011] and are created by identifying elements of the originals and relations between them. The identification of elements and their relations lead to a normalization of these originals.

We defined and related two meta-models to *integrate the multiple PRs* and one meta-model to *integrate the software project context with the multiple PRs* (Fig. 4):

- Meta-model to *normalize the structure of PRs* by selecting their elements that allow a structural integration
- Meta-model to *normalize the terminology of PRs* by relating their similar elements regarding their semantic and thus, achieving a terminology integration
- Meta-model to *model the software project context* by selecting elements for the characterization of different software project contexts and relate them to the elements of PRs

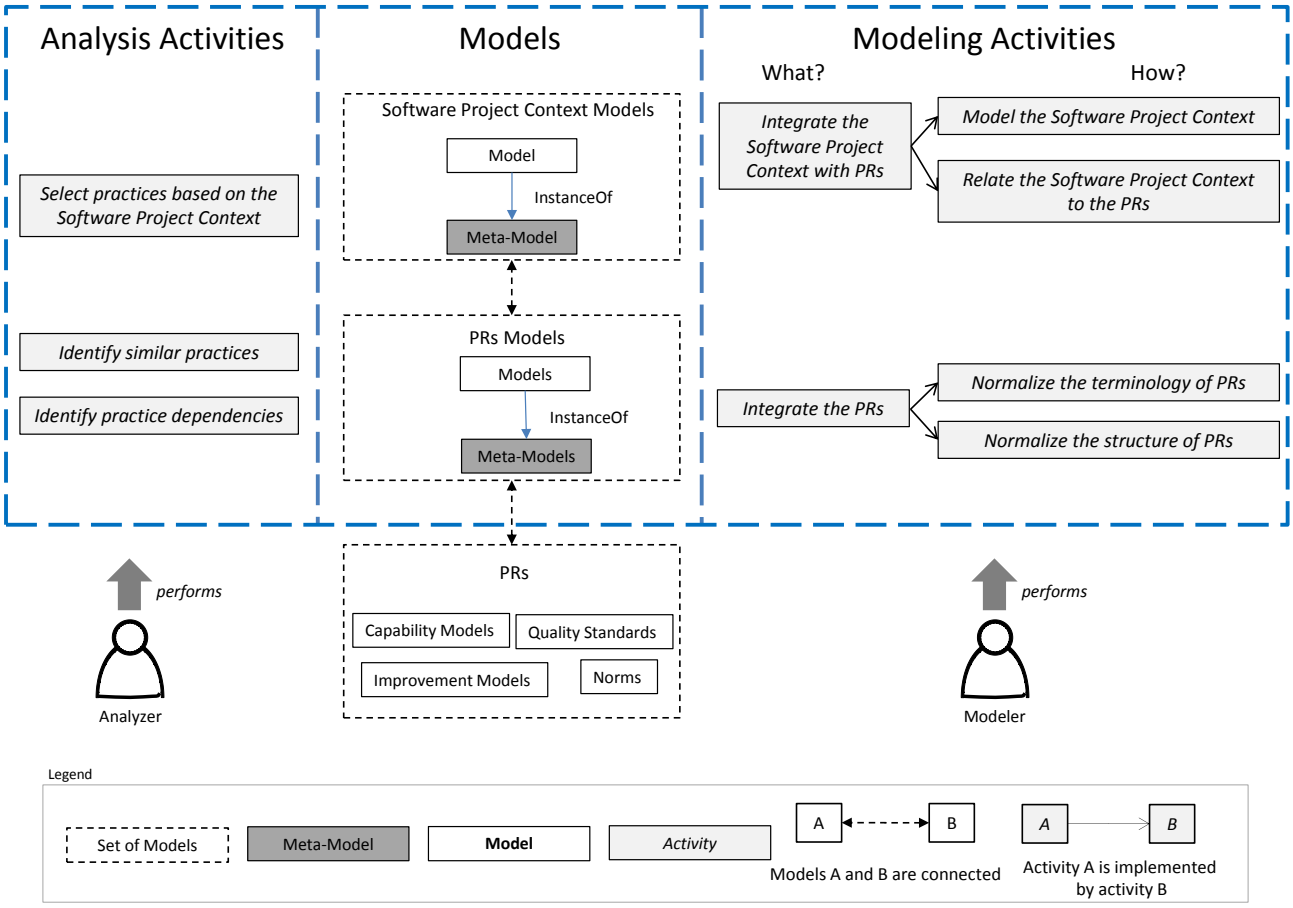


Fig. 4: MOSAIC overview

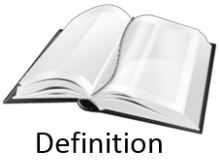
In the following, we briefly describe how we built the MOSAIC models based on ideas from the existing approaches.

2.4.1 Integration of PRs

The main purpose of the integration of PRs is to support the automated identification of similar practices and of dependencies between them (Fig. 5). This integration is performed by a Modeler.

An analysis of the related work revealed that the identification of similarities and dependencies is possible if the PRs' elements are connected. We give a detailed review of the related work in the next sub-sections.

To allow this connection between the PRs' elements, a Modeler normalizes the structure and terminology of the PRs. He constructs the **Integrated Structure Models (ISMs)** as instantiations of the Integrated Structure Meta-Model (IS Meta-Model) and the **Integrated Concept Model (ICM)** as instantiation of the Integrated Concept Meta-Model (IC Meta-Model). Consequently, the Modeler *normalizes the structure of PRs* by *extracting ISM elements*. Then, he *normalizes the terminology of the PRs* by *extracting ICM elements based on ISM elements* and *by relating these extracted ICM elements with the ISM elements* (Fig. 5).



The **extraction of ISM or ICM elements** consists of the identification of these elements in the PRs, as well as their modeling according to the guidelines defined by the MOSAIC meta-models.

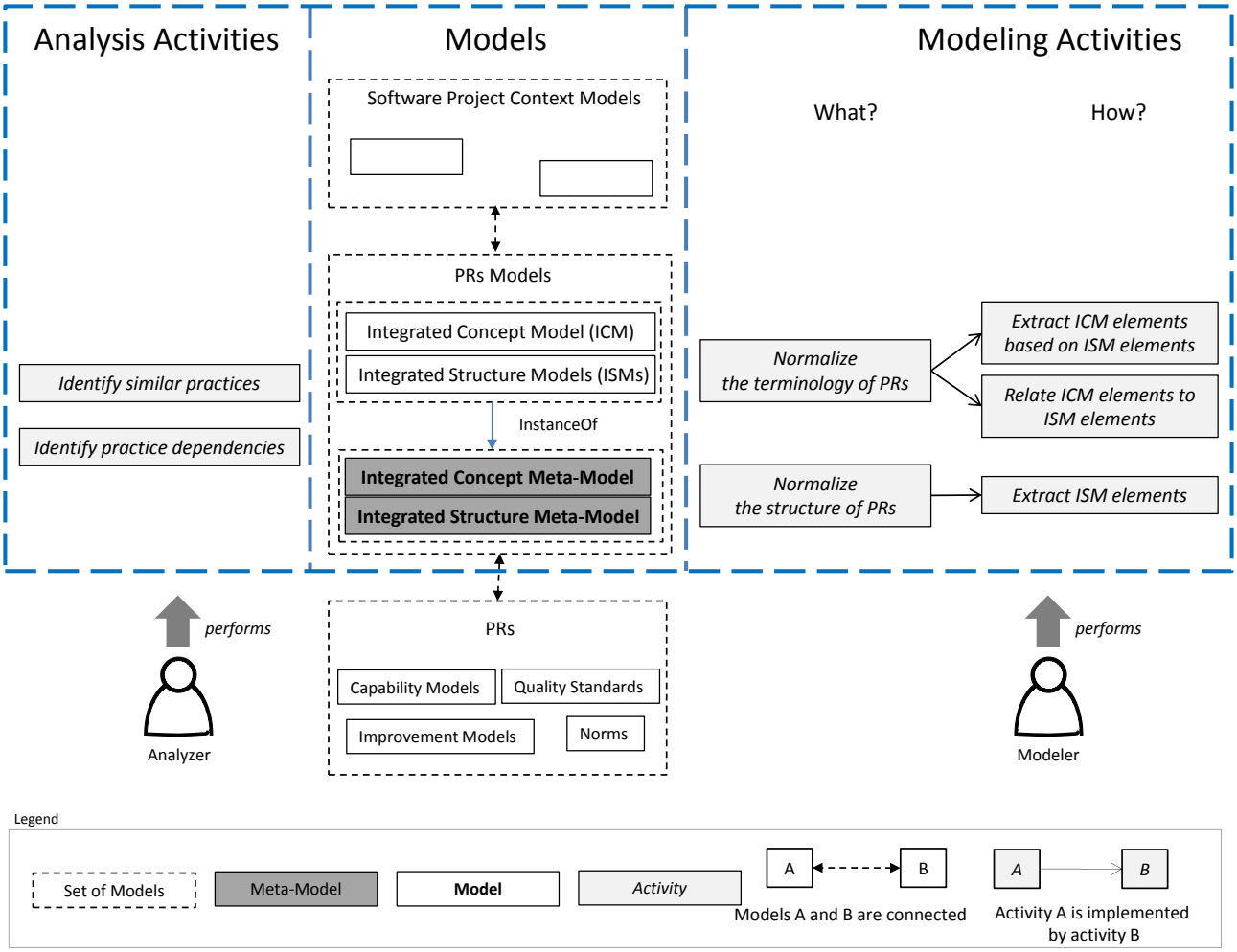


Fig. 5: Integration of PRs

2.4.1.1 Structure Normalization

To normalize the structure of multiple PRs, we defined a fine-grained common structure (IS Meta-Model) for these PRs.

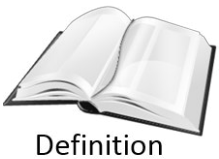
PRs have different structures. A detailed description of the meta-models of CMMI-DEV, ITIL, ISO 9001, ISO/IEC 90003, ISO/IEC/IEEE 12207 and IEEE 1028 illustrates the different structures [Kelemen 2013].

Various approaches define a common structure of PRs to compare their practices [Ferchichi and Bigand 2008; Li Liao et al. 2005; Rahm and Bernstein 2001; Wang et al. 1999]. To reveal the practice

similarities, two similar practices are manually connected in these approaches. Therefore, a bilateral comparison is supported. We aim to support a multilateral comparison of practices and allow the connection of more than two practices. Moreover, we aim to support an automated comparison of two or more practices. Finally, the related approaches do not provide any information about common and different elements of the similar practices. The comparisons are subjective as the reasoning for a mapping is not given. Yoo et al. request from such comparative studies to justify the identified mappings and warn that otherwise the results are not repeatable [Yoo et al. 2006]. The subjectivity of the mentioned comparisons is overcome by approaches based on a fine-grained structure of PRs.

The need for a fine-grained structure of PRs is mentioned in a series of articles from SEI [Siviy et al. 2008b]. Here, the practice elements, such as inputs, outputs or roles are modeled. Based on this fine-grained structure, there exist approaches that indicate the similarities and differences between practices by comparing their practice elements [Kelemen 2013; Malzahn 2009; Pardo et al. 2012; Soto and Münch 2008]. Furthermore, these fine-grained elements are also mentioned in the works related to the identification of dependencies between practices [Chen et al. 2008].

MOSAIC is also based on this fine-grained structure. We call these fine-grained elements ISM practiceConcepts.


Definition

An **ISM practiceConcept** is an ISM element, a word or a combination of words, which semantically represent one of the following practice elements:

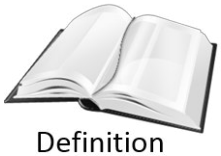
- role, which performs the activity described by the practice.
- output, which is produced by this activity.
- input, which is needed by this activity to produce the output.
- purpose, which motivates the activity operation.

The ISM practiceConcepts are the main ISM elements. A Modeler extracts these ISM practiceConcepts and other related elements from the PRs to create the ISM for each PR and thus, to normalize the different PRs.

2.4.1.2 Terminology Normalization

To normalize the terminology of multiple PRs, we defined the IC Meta-Model with its relations to the IS Meta-Model. Based on these, a Modeler creates a new model, the Integrated Concept Model (ICM), and relates it to all already created ISMs.

In the afore-mentioned related work, multiple PRs are integrated by connecting their elements. While in some approaches similar practices are connected, in others, similar practice elements are connected ([Ferchichi and Bigand 2008; Li Liao et al. 2005] vs. [Kelemen 2013; Malzahn 2009; Pardo et al. 2012; Soto and Münch 2008]). In MOSAIC, a Modeler connects practice elements, namely the ISM practiceConcepts, by introducing a new element between them. This new element relates these ISM practiceConcepts and thus, their practices become connected. We call these new elements ICM concepts.



An **ICM concept** is an ICM element, a word or the smallest combination of words contained in a practice that has a unique meaning in the context of PRs. One or more ICM concepts semantically define one ISM practiceConcept contained in different ISMs.

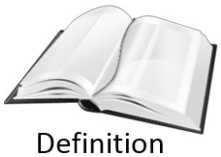
For example, “project plans”, “work breakdown structures”, “key stakeholders” or “software stakeholders” are such ICM concepts.

The ICM concepts are the main ICM elements. A Modeler extracts these ICM concepts based on ISM practiceConcepts to create the ICM, a model of all considered PRs. As a result, the PRs are indirectly connected by the central model, namely the ICM. Based on this approach with a central model, MOSAIC can be easily extended with further PRs.

As the ICM concepts connect similar ISM practiceConcepts, practices from multiple PRs can be compared. While in the afore-mentioned approaches the practice elements are connected pairwise, in MOSAIC multiple ISM practiceConcepts are connected. Therefore, a multilateral instead of a bilateral comparison of practices is possible in MOSAIC.

Another issue of the related approaches that are based on a fine-grained structure of PRs is that there is no reasoning why practice elements are connected. Moreover, there is no differentiation between various types of similarity. The connected elements can be semantically equal, can have a small, medium or high similarity. Inspired by relations from similarity theory that reflects the similarity between two concepts (such as generalization or partition mentioned in [Storey 1993]), we also defined such similarity relations to connect similar ICM concepts. Consequently, the similar ISM practiceConcepts are also connected. The different similarity relations between the ICM concepts give a reasoning for the similarity between the ISM practiceConcepts. Moreover, these similarity relations allow a Modeler to specify if the ISM practicesConcepts are equal or have a high, medium or small similarity.

The ICM concepts related by the similarity relations lead to the creation of an ontology. Hence, the ICM becomes an ontology that contains related PRs concepts. We introduce an ontology based on the definition of Thomas Gruber, who did foundational work in the ontology engineering [Gruber 1993].



An **ontology** is an explicit specification of a conceptualization, an abstract view of the world that we wish to represent for a purpose. Therefore, it is an linguistic, formal representation of a set of concepts and their relations within a particular domain.

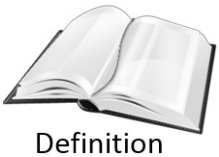
To summarize, the similarity relations between ICM concepts and the traceability of the ICM concepts back to the original ISM practiceConcepts allow an automated comparison of multiple practices and identification of practice dependencies.

2.4.2 Integration of Software Project Context with PRs

The purpose of the integration of the software project context with the multiple PRs is to support an automated selection of practices based on the software project context.

Based on an analysis of various frameworks which characterize the software project context, we selected one framework and integrated it with the multiple PRs. We give a detailed review of such frameworks in the next sub-sections.

Based on this framework, a Modeler creates the **Situational Factors Model (SFM)** as an instantiation of the Situational Factors Meta-Model (SF Meta-Model). Consequently, he *models the software project context* by *extracting the SFM elements* and by *relating these SFM elements with the ICM elements*.


Definition

The **extraction of SFM elements** consists of the identification of these elements in the selected framework that characterize the software project context, as well as their modeling according to the guidelines defined by the MOSAIC meta-models.

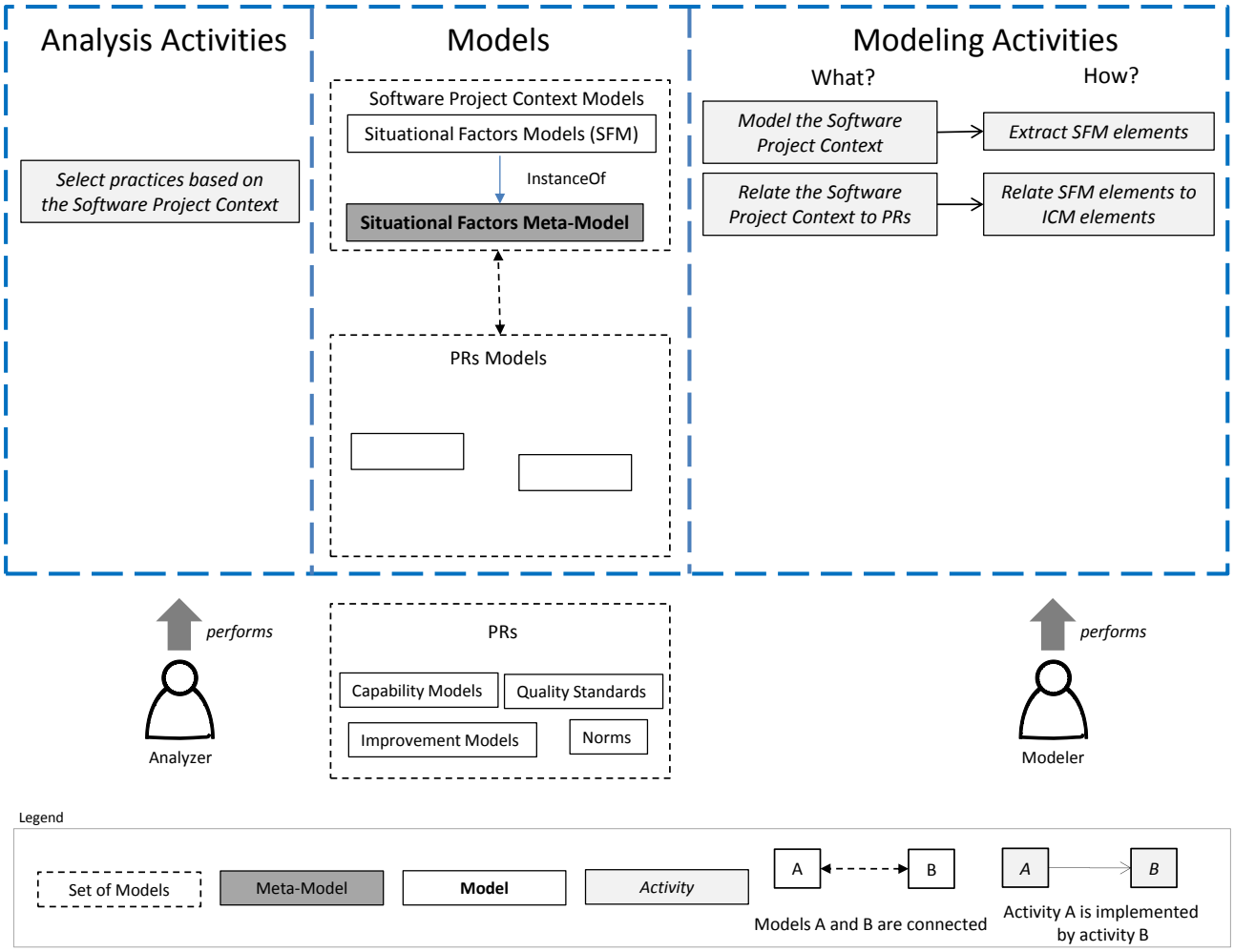


Fig. 6: Integration of the software project context with PRs

2.4.2.1 Software Project Context Modeling

To model the software project context, we selected a framework with various factors to characterize this context and create the Situational Factors Model (SFM).

We performed a literature research to identify approaches and frameworks to characterize the software project context. Estimation models, risk-based approaches or tailoring approaches define factors that describe the software project context.

Estimation models support projects in the budget estimation by using different factors that characterize the software project context. These factors describe different critical situations and are used to estimate the project costs with respect to the software project context. Models, such as COCOMO [Boehm 1981; Boehm et al. 2000] or FPA[2] [Albrecht 1979], define such factors: cost drivers or general system characteristics respectively. Some examples of factors are “complexity of the product” in COCOMO and the “performance” in FPA.

Risk-based approaches also consider the software project context as software projects can have different risks. These approaches define a list of risk factors that indicate possible critical situations

[2] FPA – information available at: www.ifpug.org (IFPUG FPA CPM v4.3.1), as well as www.cosmicon.com (COSMIC FPA Measurement Manual v3.0)

in a software project [Boehm 1991; Fairley and Willshire 2003; Keil et al. 1998; Oz and Sosik 2000]. Factors, such as "misunderstanding the requirements", "changing scope/objectives" or "inadequate skills" describe project settings that could jeopardize the software project and need to be addressed.

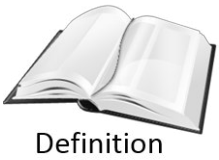
Tailoring approaches consider the software project context to adapt the internal process PRs to be used in the projects. Different types of software projects are carried out in different environments [MacCormack and Verganti 2003]. Factors to be used for tailoring approaches are defined in various frameworks [Cockburn 2000; Dede and Lioufko 2010; Kalus and Kuhrmann 2013; Petersen and Wohlin 2009; Xu and Ramesh 2007].

We decided to use the situational factors framework [Clarke and O'Connor 2012] to model the software project context. This framework is based on a profound analysis of approaches that describe the software project context. It analyzes various data sources that describe different situations in the software project context (Fig. 7). It does not only analyze all the afore-mentioned approaches (estimation models, risk-based or tailoring approaches), but also identifies factors defined by the PRs themselves (CMMI-DEV, ISO/IEC 12207 or SWEBOK [ISO/IEC TR 19759:2005 2007]). Based on this intensive analysis, a framework of situational factors was defined.

		Data Sources																							
Classification	**Factors**	Ferratt & Mai (2010)	Cameron (2002)	Benaroch & Appari (2010)	Appari & Benaroch (2010)	Wallace & Keil (2004)	Wallace, Keil & Rai (2004)	Ropponen & Lyytinen (2000)	Lyytinen, Mathiassen & Rop (1998)	Keil et al. (1998)	Barki, Rivard & Talbot (1993 & 2001)	Boehm (1991)	Casher (1984)	Boehm CoCoMo (2000)	Albrecht FPA (1984)	Putnam SLIM (1978)	Delany & Cunningham (2008)	Ropponen & Lyytinen (2000)	Xu & Ramesh (2007)	SWEBOK (IEEE 2004)	ISO-12207(2008)	CMMI(2006)	Boehm & Turner(2003)	Factors identified in review phase
Personnel	Turnover, Team Size	•				•					•		•	•					•	•				
	Culture	•																	•				•	
	Experience, Cohesion, Skill, Productivity	•	•	•	•	•	•	•		•	•		•	•		•	•		•	•		•	•	•
	Commitment					•																		
	Disharmony					•					•													
Requirements	Changeability, Feasibility	•	•		•	•	•		•	•		•	•						•		•		•	
	Standard, Rigidity		•		•	•	•	•	•	•	•	•	•	•		•	•				•			
Application	Degree of Risk	•	•			•					•		•	•										
	Performance	•		•				•				•		•	•		•							
	Complexity, Type, Size, Predictability, Connectivity			•		•	•	•	•		•			•	•	•	•	•	•	•	•		•	
	Reuse													•	•									•
	Development Phase																			•				
	Deployment Profile																							•
	Quality																			•				
Technology	Knowledge			•	•	•					•			•			•			•				
	Emergent				•	•				•	•		•	•			•							
Organisation	Maturity			•	•			•			•	•		•		•	•	•	•	•				
	Management Commitment					•				•	•						•		•					
	Stability					•	•				•								•					•
	Structure								•															
	Facilities								•												•			
	Size																			•				
Operation	End-Users					•	•			•	•		•				•							
	Prerequisites														•		•					•		
Management	Expertise					•	•	•	•	•	•	•	•	•			•	•	•			•		
	Accomplishment					•					•						•							
	Continuity					•	•										•							
Business	External Dependencies					•		•	•		•	•	•	•			•		•					
	Business Drivers																		•			•		
	Time to Market																			•				
	Customer Satisfaction																			•				
	Payment Arrangements																							•
	Opportunities																				•			
	Magnitude of Potential Loss										•		•											•

Fig. 7: Situational factors and considered data sources [Clarke and O'Connor 2012]

Accordingly, we define the SFM situationalFactors based on the definition of the situational factors.


Definition

A **SFM situationalFactor** is a situational factor defined as “a characteristic of a software development setting that is known to affect the software development"[Clarke and O’Connor 2012].

The SFM situationalFactors are the main SFM elements. A Modeler extracts them to create the SFM and thus, model the software project context.

2.4.2.2 Relation between the Software Project Context and PRs

To finalize the integration of the software project context with the PRs, we related the SF Meta-Model to the IC Meta-Model. Therefore, a Modeler relates the SFM situationalFactors to the ICM concepts.

While there are many approaches to integrate the PRs, only limited research has been done to assist software developers and managers in making decisions regarding the selection of appropriate processes or practices based on the software project context. However, there are some that mention this necessity.

Some of the afore-mentioned approaches map concrete activities to factors that characterize the software project context. The risk-based approaches define standard mitigation activities that can be adopted to decrease the risk severity for the software project [Wallace and Keil 2004]. For example, activities derived from the practices of PMBOK [Project Management Institute 2013] are defined for each risk [Tesch et al. 2007]. However, there are no approaches that consider multiple PRs and thus, the selection of best practices from multiple PRs is not addressed. Furthermore, some tailoring approaches map such factors (called tailoring criteria) to a set of abstract actions [Kalus and Kuhrmann 2013]. However, the activities are not extracted from multiple PRs.

Furthermore, there are also some approaches that propose systematic methods to map the factors to the software project context. For example, the Root-Cause-Analysis method is proposed to be used in the early stages of a software project to identify which are potential causes for software project failures [Buglione 2008]. These potential causes are characterized by different factors. Based on these causes, activities for the software process improvement can be defined [Leszak et al. 2000]. For example, an insufficient tool knowledge of the personnel can be a potential cause that can be avoided by the action “provide proper training for the personnel”. However, these approaches do not consider multiple PRs or do not define when a factor can be mapped to the considered improvement activities.

A more detailed mapping between a factor and improvement activities results in a case study with several experts. Here, the support degree between such factors and CMMI-DEV practices based on a four-point ordinal scale is defined [Jeners et al. 2013a]. However, the mapping is manually performed for each factor and each practice and considers only one PR. We aim to offer an automated mapping where for each factor, best suited practices from one or more PRs of interest are selected.

Based on the experiences gained during the afore-mentioned case study and its results, we defined relations between SFM situationalFactors and ICM concepts in MOSAIC that reflect the support degree of an ICM concept for a software project characterized by this SFM situationalFactor. Based on these relations, an automated mapping between SFM situationalFactors and practices can be performed.

2.5 MOSAIC Toolbox

As MOSAIC aims to support an automated selection of practices, as well as an automated identification of similarities and dependencies between them, we developed the MOSAIC Toolbox. This is a proof-of-concept to demonstrate that such automations are possible. The MOSAIC Toolbox is a web application that integrates a collection of tools to support a Modeler and an Analyzer to perform the modeling and the analysis activities respectively (Fig. 8).

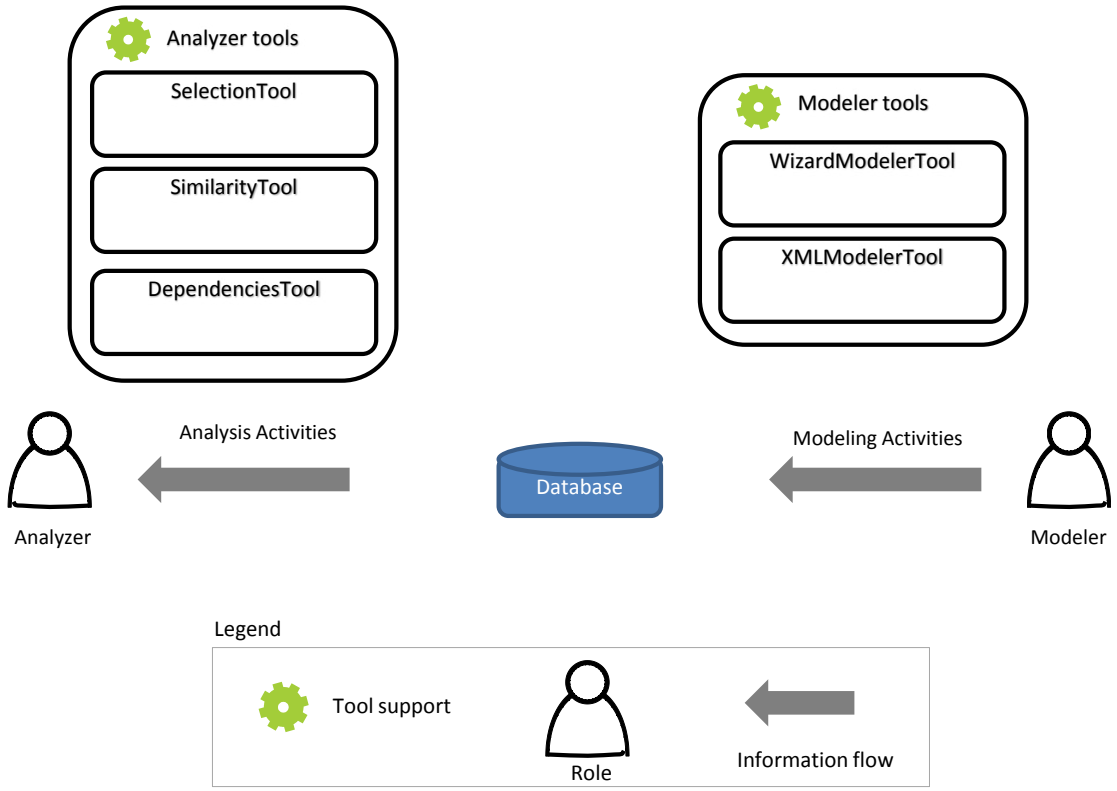


Fig. 8: MOSAIC Toolbox tools

A Modeler can use the modeler tools to model the PRs and the software project context, as well as to save the modeled data in the database. We developed the `WizardModelerTool`, that guides the Modeler to perform the modeling activities and the `XMLModelerTool`, that can be used to model the data in XML format. Depending on the particular needs of the Modeler, he chooses a certain modeler tool or uses these tools in combination for different purposes and different stages of data modeling. Therefore, we offer various tool support to model the significant amount of data.

An Analyzer uses the analyzer tools to select practices based on the software project context (`SelectionTool`), to identify similar practices (`SimilarityTool`) or to identify practice dependencies (`DependencyTool`) based on the models saved in the database.

2.6 Analyzer and Modeler Roles

In this section, we describe the MOSAIC roles and give a mapping between these roles and organizational roles.

The *Analyzer* and *Modeler* roles can be performed by various practitioners who aim to work with multiple PRs: *software process engineers*, *assessors*, *consultants* and *software project members*. There are no standard names for these roles. For example, the software process engineers are called engineering process group members [Kulpa 2003], process specialist [Buch 2010] or process engineers [Stephen 2013]. Therefore, we define them as follows:

- *Software process engineers* are responsible for the definition and improvement of the internal process PRs of an organization.
- *Software process assessors* perform assessments according to PRs at the organizational and project level.
- *Software projects members* use PRs to achieve the project goals.
- *Software process consultants* support organizations in the adoption of PRs and the assessment based on PRs.

As the software process engineers are dealing with internal process PRs, we remind about theirs definition.


Reminder

An **internal process PR** is a special process PR and refers to the internal software processes of an organization. This PR defines activities to be used as guidelines for the improvement of software processes in an organization, e.g. in software projects. Hence, it defines the practices of this organization and thus, it is a PR.

The *Modeler* role can be performed by *software process engineers* or *consultants* who have a deep theoretical and practical experience with the software processes, the considered PRs and the software project context. Their practical experience in all phases of a software project is necessary to understand how PRs can support a software project to manage important constraints and critical situations in the project. This experience supports the Modeler in the integration of the PRs with the software project context.

The *Analyzer* role can be performed by all the afore-mentioned organizational roles. We give some examples of how these roles can use the MOSAIC analysis activities.

While the software process engineers select practices based on the context of multiple software projects, the software project members select practices based on their own software project context. The selected practices are then adopted. The software process engineers improve the internal process PRs according to the selected practices. The software project members adopt the selected practices to manage their software project context and achieve the project goals. Furthermore, both roles can simultaneously use the information of multiple PRs (similarities or dependencies) to benefit from synergy effects. Some PRs are more abstract than others or give different information how to adopt a certain practice. This information can be used by the Analyzers to better understand and adopt the selected practices. Concrete examples are illustrated in the previous sections (Table 2 – Examples 3 and 4).

The software process assessors identify similar practices or the practice dependencies from multiple PRs to optimally use the multiple PRs. Consequently, redundancies or unnecessary evaluations are avoided in the assessment of the internal process PRs according to these multiple PRs. Concrete examples of redundancies and dependencies are illustrated in the previous sections (Table 2 – Example 1 and Fig. 3 respectively).

The software process consultants can also use the information about the similarities, differences and dependencies between PRs. Consequently, they can better support their customer in the software process improvement based on these PRs.

2.7 Summary

MOSAIC defines meta-models to allow a Modeler to integrate different PRs and the software project context, and an Analyzer to perform an automated selection of practices, identification of similar practices and of dependencies between them (Fig. 9).

The central model of MOSAIC is the ICM. Firstly, it allows the integration of the various PRs at a conceptual level. This integration allows the identification of similar practices and of dependencies between them. Secondly, ICM also allows for a conceptual level integration of the PRs with the software project context to select practices that are needed for addressing different situations in software development settings.

The Modeler integrates the multiple PRs by normalizing their structure and terminology. Consequently, he creates an Integrated Structure Model (ISM) for each PR and an Integrated Concept Model (ICM) for all PRs. While the structure normalization is performed by an extraction of the ISM elements, the terminology normalization is performed by an extraction of ICM concepts based on ISM practiceConcepts and the relation of these extracted ICM concepts to ISM practiceConcepts.

The Modeler integrates the multiple PRs with the software project context by modeling this context and relating it to the PRs. Consequently, he creates the Situational Factors Model (SFM) and relates it to ICM. The modeling of the software project context is performed by an extraction of the SFM situationalFactors from an existing framework [Clarke and O'Connor 2012]. Finally, based on the experiences gained during a case study performed by a group of experts to map SFM situationalFactors to CMMI-DEV practices, we defined relations between SFM situationalFactors and ICM concepts. The Modeler instantiate these relations and thus, relates the SFM with the ICM.

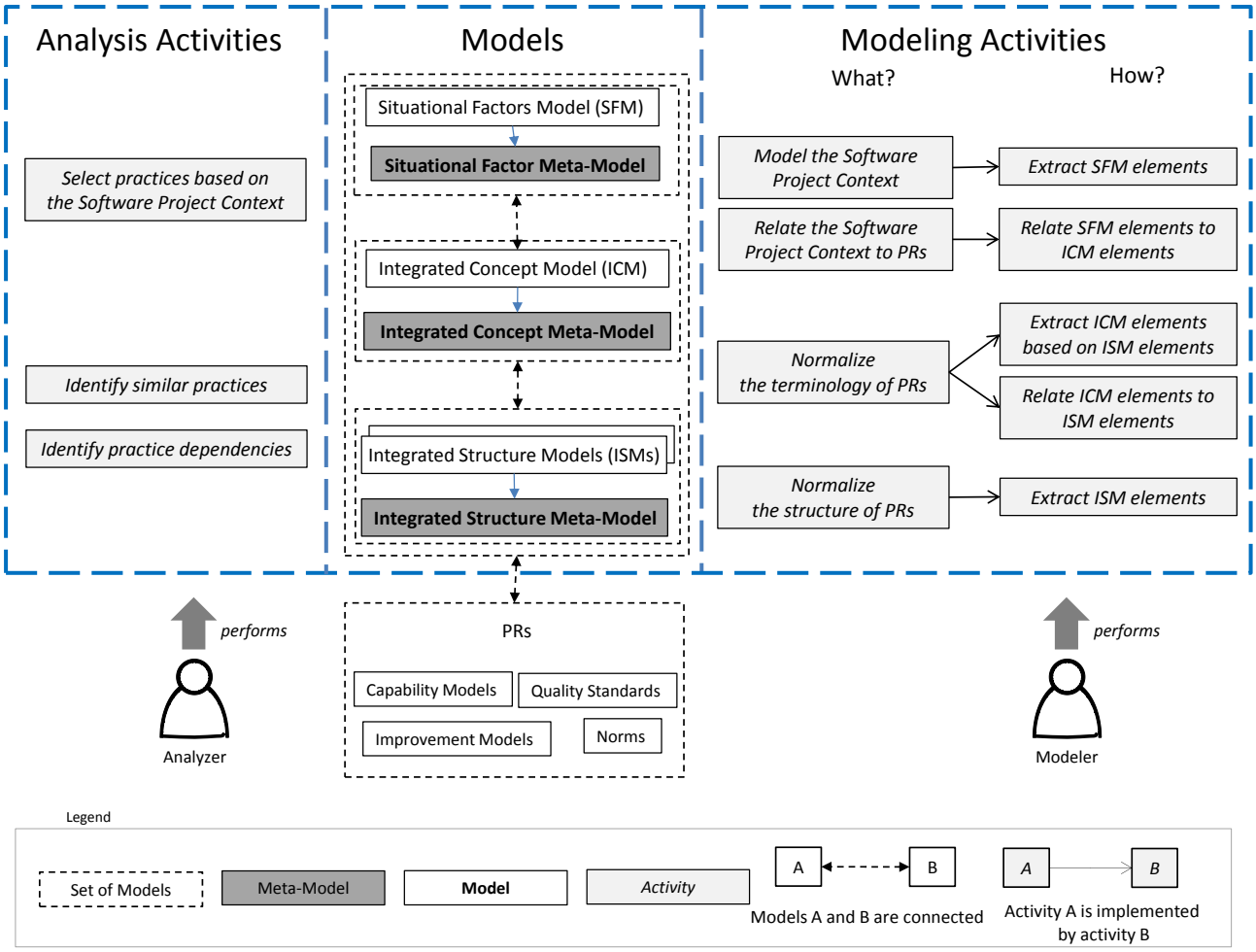


Fig. 9: Integration of multiple PRs and of software project context

Table 5 gives an overview of the goals mentioned in chapter 1, the working fields that need to be addressed to achieve these goals and finally, an overview of the MOSAIC way to solve these working fields.

Table 5: Goals, working fields and overview of the proposed solution

Goals	Working fields	Overview solution
(G1): Automatically select practices from multiple PRs based on the software project context.	(WF1.1.1): Analyze various frameworks that describe the software project context.	Various frameworks define lists of factors to be used for the description of the software project context.
	(WF1.1.2): Select a framework that characterizes the software project context.	The situational factors framework [Clarke and O'Connor 2012] defines a list of factors that describes the characteristics of the software project context.
	(WF1.2): Develop an approach to connect the selected framework that characterizes the software project context to practices of considered PRs.	The approach is based on the extraction of SFM situationalFactors and their relation to ICM concepts.
	(WF1.3): Develop an approach to systematically and automatically select practices based on the software project context.	The automated approach is based on the computation of the support degree of practices for a certain software project context.
(G2-3): Automatically identify similar practices and dependencies between practices from multiple PRs.	(WF2-3.1.1): Analyze various related approaches to integrate the multiple PRs and allow the identification of similar practices.	Various approaches connect the PRs' elements at the practice or at the practice elements level.
	(WF2-3.1.2): Provide a flexible design to integrate other PRs for other software areas.	For each PR, its corresponding ISM is created. The created ISMs are connected by the ICM as this contains all concepts of the considered PRs.
	(WF2-3.1.3): Analyze and offer different tool support to integrate the significant amount of data.	The MOSAIC Toolbox offers different modalities to model the PRs.
(G2): Automatically identify similar practices from multiple PRs.	(WF2.2.1): Analyze similarity theory to identify when objects are similar and how can we identify their similarity.	Objects are similar when their features are similar. Therefore, practice are similar when their elements are similar.
	(WF2.2.2): Identify different modalities to compare two or more practices and identify their similarity.	Different modalities to identify similar practices are proposed: identification of the similarity, coverage and output states of practices.
	(WF2.2.3): Develop an approach to systematically and automatically identify similar practices.	The afore-mentioned modalities to identify similar practices are implemented by automated approaches. These are based on the computation of the similarity degree, coverage degree and output states
(G3): Automatically identify dependencies between practices from multiple PRs.	(WF3.2.1): Develop an approach to connect practices that are dependent.	The identification of the dependencies between practices is based on the relation between their inputs and outputs.
	(WF3.2.2): Develop an approach to systematically and automatically identify the dependencies between practices.	The automated approach is based on the computation of the dependency degree of practices.

3 Meta-Models

MOSAIC defines three meta-models, the **Integrated Structure Meta-Model** (IS Meta-Model), the **Integrated Concept Meta-Model** (IC Meta-Model) and **Situational Factor Meta-Model** (SF Meta-Model). The first two meta-models are used to normalize and integrate the structure and the terminology of different PRs. The last one is used to model the software project context and to integrate it with the PRs. In the following sections, we describe the three meta-models with their elements, attributes and relations.

3.1 Integrated Structure Meta-Model

The purpose of the Integrated Structure Meta-Model (IS Meta-Model) is to normalize the structure of various PRs and to integrate them. Its elements (called ISM elements) and their relations are grouped in three packages: Practice Repositories, Practice and Practice Language (Fig. 10).

A model is a representation of a natural or artificial original (abstraction). It does not reproduce all the aspects of the original, but only the ones necessary (reduction) to achieve certain goals (pragmatism) [Stachowiak 1973]. Therefore, we define the IS meta-model by selecting only elements from the PRs needed to satisfy our goals. Our goal is to allow an automated identification of similar practices and of dependencies between them. Therefore, practices and related elements have to be contained in the IS Meta-Model.

Package *Practice Repositories* contains elements (called ISM practice repository elements) that group the practices of the multiple PRs. Such elements need to be modeled, otherwise an ISM of a PR would contain a long list of practices and thus, it would be difficult to work with. The ISM practice repository elements are mostly defined by the meta-models of existing PRs. As these elements have different names, we define the ISM practice repository elements to normalize their names. Table 6 exemplifies the mapping between the ISM practice repositories elements and the corresponding elements from several PRs.

ISM practice repository elements	PracticeRepository	CMMI Constellation	SPICE	IEC 61508	ISO 12207	ITIL	COBIT
	Category	Category	Process category	Part	Process category	Book	Domain
	Process	Process area	Process	Safety lifecycle phase	Process title	Process	Process
	Practice	Goal, Practice	Best practice	Objectives, Requirements	Activity	Process activity	Control objective, Control practice

Table 6: Mapping ISM practice repository elements to PRs meta-model elements – Examples

A **PracticeRepository** represents a certain PR and is structured by means of **Categories**. A Category defines a certain topic that is addressed in one or more **Processes**. A Process addresses a topic to be improved by defining **Practices**. For example, we found high similarities between COBIT control objectives, control practices, CMMI goals, practices (inclusively sub-practices), SPICE best practices or IEC 61508 objectives, requirements and thus, we consider all these mentioned elements as Practices. Finally, Processes and Practices have an **id**. For a better understanding, we give some examples for these elements. The PracticeRepository "SPICE" is structured by means of the Category

"Management". A certain topic within this Category is addressed in the Process "Project Management" with the id SPICE MAN.3. This Process contains the Practice "Implement planning activities of the project" with the id SPICE MAN.3 BP9.

Package *Practice* defines elements (called ISM practice elements) to model the PRs' information on a fine-grained level. These elements are the basis for the automated analysis activities and thus, we need to model them. We define these elements based on a literature review. Various process architectures define elements that are used to describe processes [Bhuta et al. 2005; Curtis et al. 1992; Kellner et al. 1999]. Furthermore, approaches that model the PRs on a fine-grained level define elements to describe practices [Kelemen 2013; Malzahn 2009; Pardo et al. 2012; Siviy et al. 2008b; Soto and Münch 2008].

ISM practice elements	(Curtis et al., 1992)	(Kellner et al., 1999)	(Bhuta et al., 2006)
Activity	Process step	Key activities, Tasks	Project activity
Output	Artefact	Primary objects	Output
Input	Artefact	Primary objects	Input
Role	Agent, Role	Vital resources	-
Purpose	-	-	-

Table 7: General process architectures – Process elements – Examples

ISM practice elements	(Siviy et al., 2008)	(Soto and Münch, 2008)	(Malzahn, 2009)	(Pardo et al., 2012)	(Kelemen, 2013)
Activity	-	Activity	Activity	Activity	Activity
Output	Output	Product	Outgoing work product	Product	Artifact
Input	Input	Product	Incoming work product	Product	Artifact
Role	Role, Responsability	Role	Stakeholder	Ressource	Role, Responsability
Purpose	-	-	-	Objective	-

Table 8: Related Work – Practice elements – Examples

There are different types of ISM practice elements: **Activities, Outputs**, **Inputs**, **Roles** and **Purposes**. An Activity with its related Outputs, Inputs, Roles and Purposes are grouped in an **ActivityUnit**. Activities need or produce Artifacts (Inputs or Outputs), have Purposes or involve Roles. For example, in the CMMI-DEV PP SP1.1 Practice "Establish a work breakdown structure to estimate the scope of the project" there is one ActivityUnit that groups the Activity and its related Outputs and Purposes. This Activity "establish the work breakdown structure" produces the Output "work breakdown structure" and has the Purpose "to estimate the scope of the project". The Activity does not need any Inputs and does not involve any Roles.

Furthermore, there are **explicit Artifacts** (**isExplicit=true**) and **implicit Artifacts** (**isExplicit=false**). The explicit Artifacts are contained in a Practice and can be directly derived from its text. The implicit Artifacts are contained in the PRs' description of a Practice or are defined in the original meta-model of PRs. For example, the CMMI-DEV defines examples of typical work products and SPICE defines outcomes for a Practice that can be considered as implicit Artifacts.

Finally, **PracticeConcepts** are ISM practice elements that are related to Concepts in the IC Meta-Model (details in next section). Inputs, Outputs, Roles and Purposes are such PracticeConcepts.

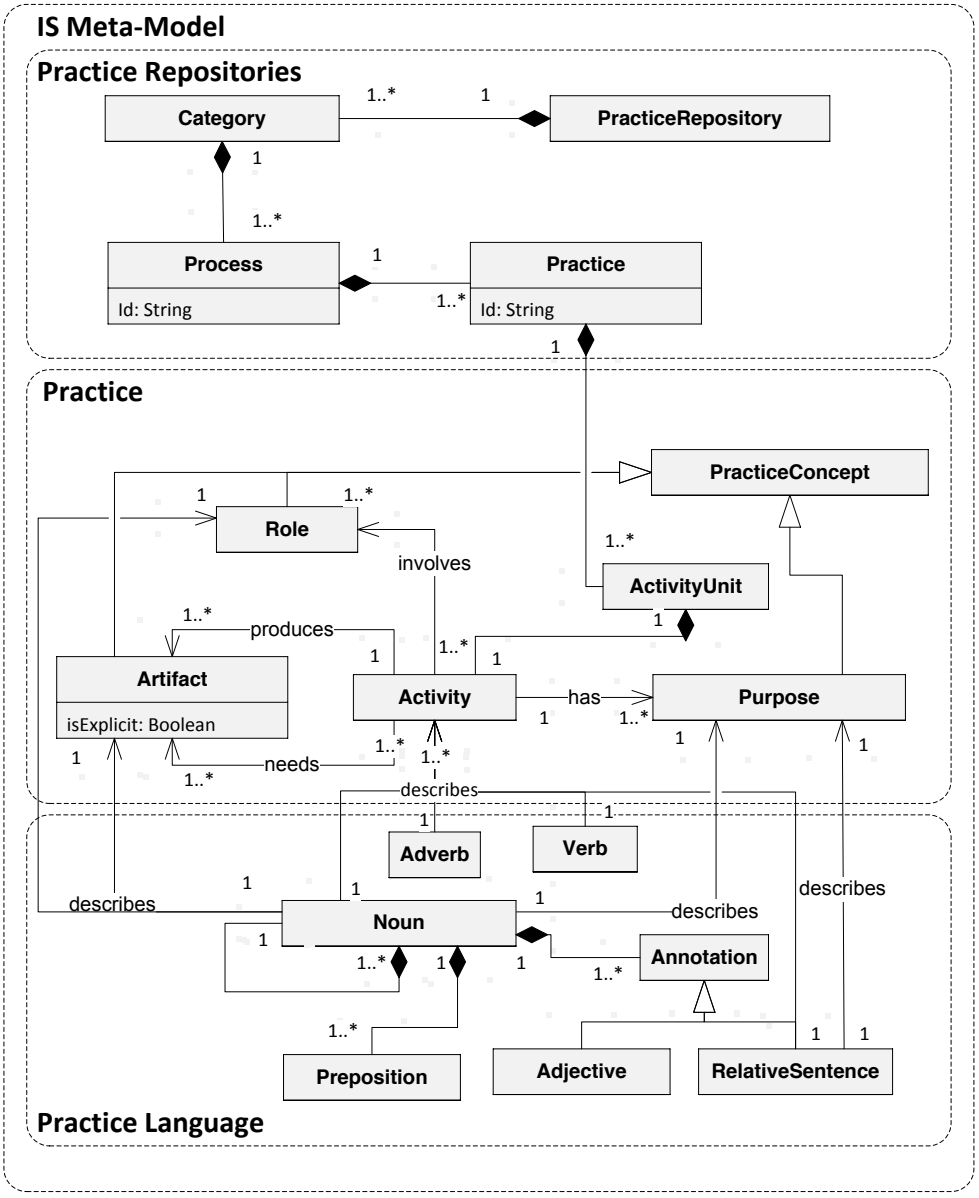


Fig. 10: Integrated Structure Meta-Model

Package *Practice Language* defines syntactical elements (called ISM practice language elements) that are used to textually describe the ISM practice elements. The purpose of these elements is to normalize the language of the ISM practice elements as each PR has its specific language style. For example, some PRs use verbs in their active form, while others PRs make extensive use of gerunds or nominalizations for the description of activities.

Based on a syntactical analysis of the language of ISM practice elements, we define for each ISM practice element its corresponding ISM practice language element:

- An Artifact and a Role are described by a Noun. A Noun can contain one or more Prepositions and build composed Nouns (e.g. "records of quality assurance activities"). A Noun can be specialized by the use of one or more Adjectives (e.g. "formal practice") or RelativeSentences (e.g. "organizational structure that reflects business needs").

- An Activity can be described by a Verb and Noun (e.g. "create a supplier agreement") or by a Verb and a RelativeSentence (e.g. "verify that personnel have the competencies").
- A Purpose can be described by a RelativeSentence (e.g. "use effective methods to package the assembled product") or by a Noun introduced by a Preposition (e.g. "deliver with confirmation").

Different steps are performed to achieve the current state of the IS meta-model [Jeners (Pricope) and Lichter 2011; Jeners et al. 2012b; Jeners et al. 2012c; Jeners et al. 2012a; Jeners and Lichter 2013; Jeners et al. 2013c]. Activities, Inputs, Outputs and Roles are the main elements of MOSAIC and thus, they remained stable over several iterations performed to improve the IS Meta-Model. Other elements are removed (e.g. OrganizationsLevel, QualityAttribute) or renamed (e.g. Context to Purpose) to achieve an IS Meta-Model that better serves its purposes. Furthermore, the IS Meta-Model was extended with the ISM practice language elements as we realized that further guidelines are needed to consistently identify the ISM practice elements.

3.2 Integrated Concept Meta-Model

The purpose of the Integrated Concept Meta-Model (IC Meta-Model) is to normalize the terminology of the different PRs and to integrate them. Its elements are called ICM elements (Fig. 11).

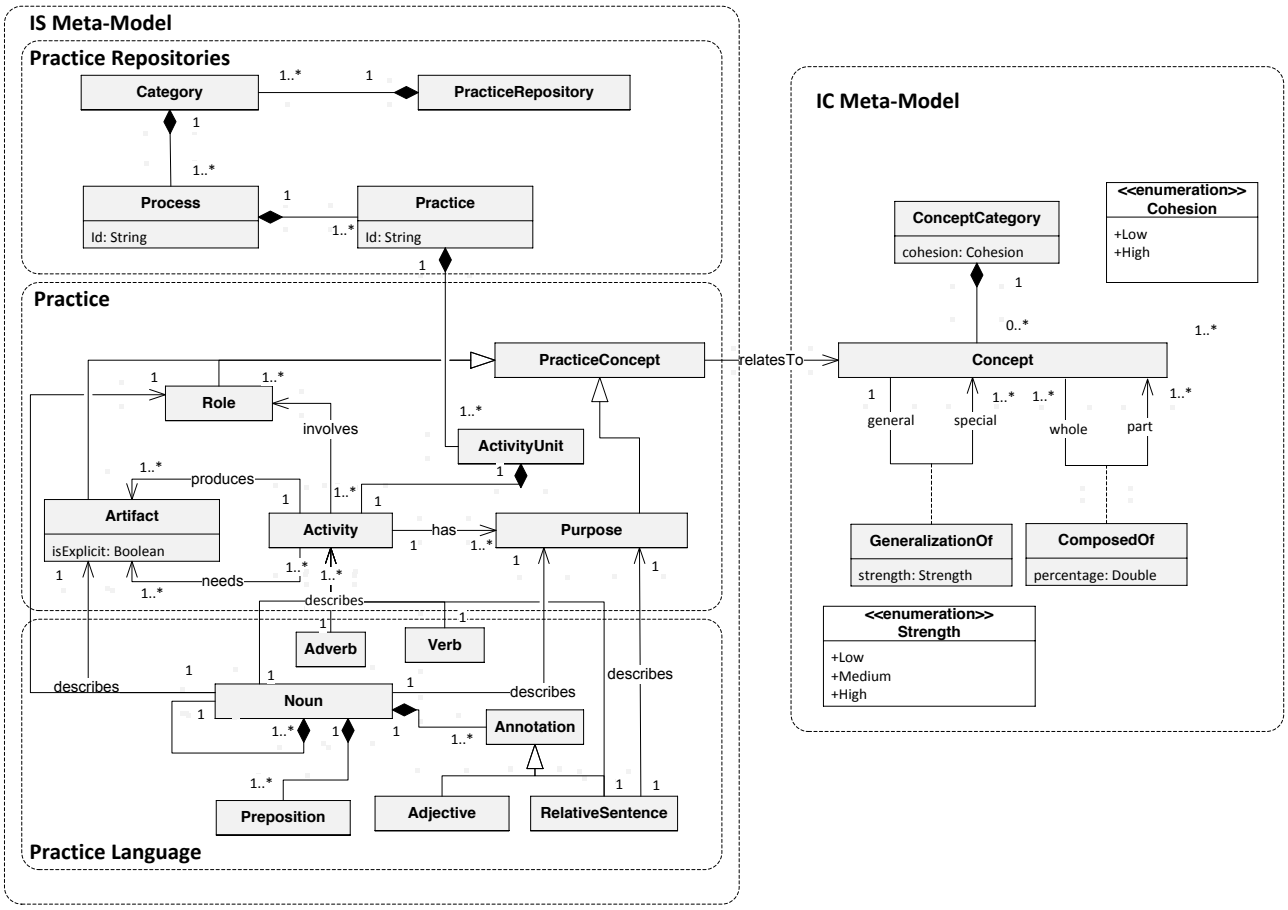


Fig. 11: Integrated Structure and Concept Meta-Models

A **Concept** is essential for MOSAIC as it allows the integration of PRs with each other and of PRs with the software project context.


Reminder

An **Concept** is a word or the smallest combination of words contained in a practice that has a unique meaning in the context of PRs.

The modeling of Concepts leads to an ontology of the terms used in the context of PRs, namely to the ICM.

According to Rector, who defines guidelines for the construction and maintenance of ontologies, an ontology should not contain aggregated concepts [Rector 2002]. Aggregated concepts lead to a cluttered ontology, an ontology that cannot be maintained any more due to the high number of elements and relations. As our aim is to create a maintainable ICM and thus, avoid a cluttered ICM, we define a Concept as a word or as the smallest combination of words, but not as an aggregation of such Concepts. For example, the "software key stakeholders" is an aggregation of the Concepts "key stakeholders" and "software stakeholders" and thus, it is not a part of the ICM.

Rector proposes to define aggregated concepts outside the ontology and relate them with their one or more semantically corresponding non-aggregated concepts inside the ontology. Therefore, MOSAIC contains the PracticeConcepts (aggregated concepts) in the IS Meta-Model that are related (**relatesTo**) to one or more semantically corresponding Concepts (non-aggregated concepts) in the IC Meta-Model.

Although PRs have redundancies, there exist a high number of Concepts when integrating the multiple PRs. To avoid a cluttered ICM, we propose to structure the Concepts. Based on the ideas of facet-classification for similar software components [Schmidt et al. 2010], role modeling for model management [Kensche et al. 2007] and "playable roles" for concepts in ontologies [Rector 2002], we define **ConceptCategories** to categorize the Concepts. For a ConceptCategory, we define its cohesion with the values "Low" and "High". This reflects how close its Concepts are to each other, i.e. reflects the similarity between these Concepts. ConceptCategories do not have to be defined for all Concepts.

Different PRs have similarities and thus, their Concepts are similar. We define similarity relations to connect Concepts and thus, to support the identification of similarities and dependencies between Practices. There exist various similarity relations between concepts in an ontology [Storey 1993]: the **generalization** relation ("*an entity type is the union of non-overlapping subtypes*") and the **partition** relation *("a set of members is considered an object in its own right")*. Based on these notions, we define the **GeneralizationOf** and **ComposedOf** similarity relations. For example, the Concept "project plans" is a GeneralizationOf "software project plans". The Concept "project plans" is ComposedOf "activities", "roles" and "activities dependencies".

As there can be similarity relations between Concepts, we define the following notions:

- **Similar Concepts** are related by ComposedOf or GeneralizationOf.
- **Different Concepts** are not related by ComposedOf or GeneralizationOf.
- **Synonym Concepts** have the same sematic meaning and thus, they are semantically equal and refer to the same Concept.

The PRs for a certain software area can have high similarities and thus, their Concepts are to a high extent similar. Therefore, there can exist a high number of similarity relations between these

Concepts. This could lead to a cluttered ICM. Ontologies have to be maintained by restricting the definition of similarity relations between concepts [Lembo et al. 2006].

Firstly, we define hierarchies for GeneralizationOf and ComposedOf.

The usage of tree hierarchies lead to maintainable ontologies [Rector 2002; Welty and Guarino 2001]. Based on this idea of tree hierarchies, the GeneralizationOf relates one **general Concept** with one or more **special Concepts**. Therefore, mono hierarchies with general and special Concepts are formed. We call them **generalizationOf-mono-hierarchies.** The usage of generalizationOf-mono-hierarchies decreases the number of relations GeneralizationOf between Concepts and thus, avoids the creation of a cluttered ICM. Based on notions from the graph theory, we use the following terms to differentiate between different roles of general and special Concepts inside a generalizationOf-mono-hierarchy:

- **Abstract Concept** is a general Concept and is the root of the generalizationOf-mono-hierarchy.
- **Parent Concept** for a special Concept is its general Concept for a GeneralizationOf.
- **Child Concept** for a general Concept is its special Concept for a GeneralizationOf.
- **Sibling Concepts** are special Concepts that share the same general Concept for a GeneralizationOf.
- **Ancestor Concept** for a Concept *conc* is a general Concept on the single path between the Concept *conc* and the root of the generalizationOf-mono-hierarchy.
- **Descendant Concept** for a Concept *conc* is a special Concept on the many paths between the Concept *conc* and Concepts at lower levels in the generalizationOf-mono-hierarchy.

The usage of ComposedOf also leads to the formation of hierarchies. ComposedOf relates one or more **whole Concepts** with one or more **part Concepts**. Therefore, poly hierarchies with whole and part Concepts are formed. We call them **composedOf-poly-hierarchies**. Analogously to the generalizationOf-mono-hierarchies, the whole and part Concepts inside a composedOf-poly-hierarchy can have different roles. We do not define them as we do not make use of these terms in this work.

Secondly, generalizationOf-mono-hierarchies can be connected only by one relation ComposedOf. There can be more relations ComposedOf between the Concepts in different generalizationOf-mono-hierarchies. For example, ComposedOf relates the Concepts "project plans" and "activities", as well as their child Concepts "development project plans" and "development activities". This kind of relations leads to a cluttered ICM. Therefore, we restrict the definition of the similarity relations ComposedOf. We generally[3] allow only single-point connection between the generalizationOf-mono-hierarchies, i.e. their abstract Concepts can be connected by ComposedOf. The relations ComposedOf are implicitly valid for their descendant Concepts.

For the relations GeneralizationOf and ComposedOf, we define an attribute to reflect the similarity between the related Concepts:

- **Strength** for GeneralizationOf has the values "Low", "Medium" and "High"**.** For example, "stakeholders" is a GeneralizationOf "key stakeholders" and "programmers", but the similarity between "stakeholders" and "key stakeholders" is higher than between "stakeholders" and "programmers".
- **Percentage** for ComposedOf has ratio values. For example, "project plans" is ComposedOf "activities" with a higher percentage as it is ComposedOf "roles" and "activities dependencies".

[3] There is an exception, i.e. when an abstract Concept is too general to be connected to another abstract Concept (e.g. not every Concept "*plans*" is composedOf "*project lifecycle phases*"; the Concept "*project plans*" is, but the Concept "*review plans*" is not).

To summarize, ICM with its Concepts related by GeneralizationOf and ComposedOf represents an ontology that is built according to different modeling guidelines to avoid a cluttered ontology. These modeling guidelines are necessary to perform a maintenance and evolution in a structures and systematic way [Baader et al. 2007]. Hence, guidelines, such as the definition of non-aggregated Concepts, categorization of Concepts, the definition of the generalizationOf-mono-hierarchies and composedOf-poly-hierarchies help a Modeler to obtain a maintainable ICM.

3.3 Situational Factors Meta-Model

The purpose of the Situational Factors Meta-Model (SF Meta-Model) is to model the software project context, to relate it to the PRs and thus, to integrate them. Its elements are called SFM elements (Fig. 12).

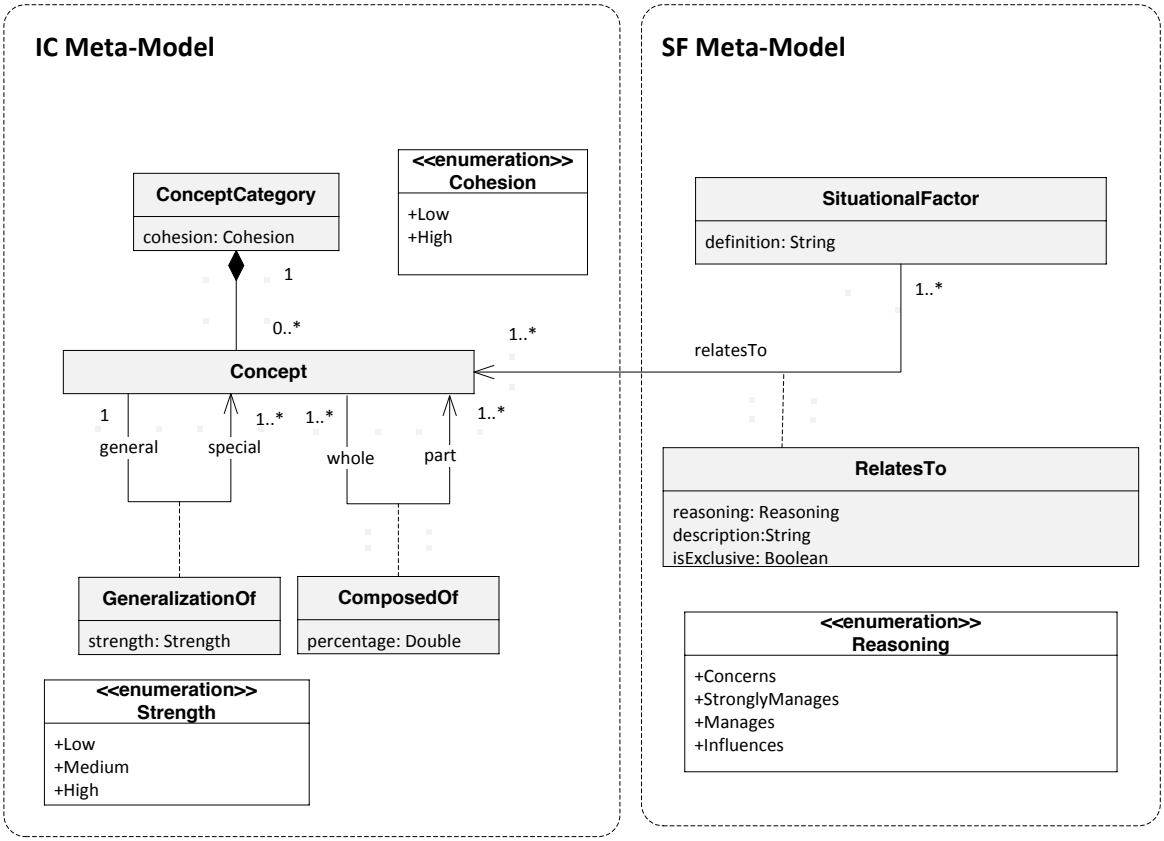


Fig. 12: Integrated Concept and Situational Factors Meta-Models

A **SituationalFactor** characterizes the software project context. As already mentioned in 2.4.2.1, there are various frameworks that define factors to characterize the software project context. As the situational factors framework [Clarke and O'Connor 2012] defines a comprehensive list of such factors, the SFM situationalFactors are the situational factors proposed by this framework (Fig. 13).

		Description
Classifications	Personnel	Constitution and characteristics of the non-managerial personnel involved in the softare development efforts
	Requirements	Characteristics of the requirements
	Application	Characteristics of the application(s) under development
	Technology	Profile of the technology being used for the software development effort
	Organisation	Profile of the organisation
	Operation	Operational considerations and constraints
	Managment	Constitution and characteristics of the software development management team
	Business	Strategic and tactical business consideration
		Sub-Factors
Personnel	Turnover	Turnover of personnel
	Team Size	(Relative) team size
	Culture	Team culture / Resistance to change
	Experience	General experience / Team experience / Team diversity / Team Ability to understand the human implications of a new information system / Team Ability to work with top management / Application experience / Analyst experience / Programmer experience / Tester experience / Experience with development methodology / Platform experience
	Cohesion	General cohesion / Team members who have not worked for you / Team not having worked together in the past / Team ability to successfully complete a task / Team ability to work with undefined elements and uncertain objectives / Overdependence on team members / Distributed team / Team geographically distant
	Skill	Operational knowledge / Team expertise (task) / Team Ability to work with undefined elements and uncertain objectives / Training development team members / Expectation of personnel's abilities / Analyst capability / Programmer capability / Tester capability / Team understanding of application
	Productivity	Team ability to carry out tasks quickly / General productivity
	Commitment	Commitment to the project among team members
	Disharmony	Interpersonal conflicts
Req's	Changeability	Scope creep / Continually changing system requirements / Ill-defined project goals / Gold plating / Unclear system requirements
	Feasibility	Straining computer-science capabilities
	Standard	General quality of input and output requirements / Conflicting system requirements / Incorrect system requirements / Misunderstanding of the requirements / Engagement of end-users in requirements capture / User understanding of requirements
	Rigidity	Rigidity of compliance to requirements
Application	Degree of Risk	Number of people and departments that the project affects / Thoroughness of design and risk resolution / Application causes major changes to the way end-users work
	Performance	Evaluation of performance requirements / Real-time performance shortfalls / Required reliability / Estimation of hardware/software capabilities
	Complexity	Product complexity / Hardware architecture / Task complexity
	Type	Application type / Application domain / Application criticality / Architecture type / Configuration demands / Back up and recovery demands
	Application Size	Hardware / Software / Required storage / Relative project size and duration
	Predictability	Extent of recent changes / Platform volatility
	Connectivity	Number of links to existing systems / Number of links to future systems
	Reuse	Required reuse / Extent of utilisation of externally-sourced components
	Development Phase	Development / Maintenance / Other phases (e.g. end of life)
	Deployment Profile	Number of deployed versions of applications / Number of deployed applications
	Quality	Required product quality / Maintainability
Tech.	Knowledge	Language experience / Tools experience / Experience with (general) technology / Project involves use of technology that has not been used in prior projects / Introduction of new technology (to general solutions base) / Need for new hardware/software
	Emergent	Emerging technology (the technology itself is emergent)
Organisation	Maturity	Maturity of the organisation / Use of modern programming practices / Availability of technical support
	Management Commitment	Senior management commitment to project / Lack or loss of organisational commitment to project
	Stability	Resources shifting from the project due to changes in organizational priorities / Unstable organizational environment / Affect of corporate politics on projects / Organization undergoing restructuring during the projects / Rate of organisational change (growth or decline)
	Structure	Organisational structure
	Facilities	Physical working arrangements / Facilities to house the project
	Organisation Size	Size of the organisation
Oper.	End-Users	Users resistant to change / End-user commitment / Degree of user engagement with development team / Conflict between users / Conflict between users/departments / Number of users outside the organisation / Number of users in organisation / Number of hierarchical levels occupied by end-users / User turnover / End-user experience with the activities to be supported by the future application / End-user familiarity with application type / End-user understanding of system capabilities and limitations
	Prerequisites	Applicable standards / Applicable laws / Organisational policies / Common practices / Operational ease / Installation ease
Management	Expertise	Effectiveness of project management methodology / Project planning capability / Project management systems / Experience with project management tools / Efficiency of governance structure / Appropriateness of rewards / Project sizing capability / Achievability of schedules and budgets / Estimation capability with respect to the personnel needs of projects / Degree of people skills in project leadership / Progress control capability / Effectiveness of work flow and coordination / Definition of project milestones / Effectiveness of project managers / Management communication skills / Manager familiarity with team / Effective understanding of responsibilities / User expectation management capability
	Accomplishment	Project management experience / Operational knowledge of leader
	Continuity	Changes in organisational management
Business	External Dependencies	Dependency on outside suppliers / Number of hardware suppliers / Number of software suppliers / Multiple implementers / Multisite development / Number of involved parties / Reliance on other projects or processing systems / Number of (external) stakeholders / Stakeholders' background / Access to Stakeholders
	Business Drivers	Project drivers / Finance considerations / Marketing activites / Maximise profit/turnover / Minimise costs
	Time to Market	Time to Market
	Customer Satisfaction	Customer satisfaction / Satisfaction with user interface
	Payment Arrangements	Time and Materials / Fixed price / Non-conventional payment arrangement
	Opportunities	Project opportunities
	Magnitude of Potential Loss	Customer relations / Financial health / Competitive position / Organisational reputation/survival / Market share / Loss of human Life

Fig. 13: Situational factors framework [Clarke and O'Connor 2012]

In this framework, the situational factors are categorized. For example, the category *"*technology*"* describes the technology profile being used for the software development. The category "requirements" describes the different characteristics of requirements for a software project. The SF

Meta-Model does not contain such a category as the number of SituationalFactors is small and their categorization is not necessary for our purposes.

Each SituationalFactor has to be defined. For each situational factor, the situational factors framework [Clarke and O'Connor 2012] lists its sub-factors. However, we observed in a case study (section 8.4.2) that the definition of a situational factor by its sub-factors is not enough. Therefore, we require a **definition** for a SituationalFactor.

One or more SituationalFactors are related to one or more Concepts of the IC Meta-Model (**relatesTo)**. We say, there is a relation between SituationalFactors and Concepts. If there is a relation between a SituationalFactor and a Concept, then this relation is also valid for all the descendant Concepts of this Concept.

Furthermore, based on the results of the afore-mentioned case study of mapping CMMI-DEV practices to SituationalFactors, we define the **reasoning** for a relation between a SituationalFactor and a Concept. It can have the following values:

- **Concerns**: The software project context characterized by the SituationalFactor concerns the adoption of the Concept. For example, the "requirements changeability" concerns the adoption of the Concept "change requests".
- **StronglyManages**: The software project context characterized by the SituationalFactor is strongly managed by the adoption of the Concept. For example, the "requirements changeability" is strongly managed by the adoption of the Concept "analyzed change requests".
- **Manages**: The situation characterized by the SituationalFactor is managed by the adoption of the ICM concepts. For example, the "requirements changeability" is managed by the adoption of the Concept "traceability matrix".
- **Influences**: The situation characterized by the SituationalFactor influences the adoption of the Concept. For example, the "requirements changeability" influences the adoption of the Concept "project plans" and the software project members have to pay attention about the impact of this situation in the project.

The adoption of a Concept to handle a certain situation in a software project context refers to the adoption of the Inputs and Outputs that are related to this Concept.

Furthermore, we define the attribute **description** for the RelatesTo association to offer the possibility to give a detailed reasoning for the relation between a SituationalFactor and a Concept.

Finally, the relation between a SituationalFactor and a Concept can be **exclusive**. This attribute is true, when exactly this combination of related Concepts have to be adopted, i.e. only the Inputs and Outputs that are related to all these Concepts have to be adopted. For example, if there is an exclusive relation between a SituationalFactor and the Concepts "committed requirements" and "development requirements", then only Inputs or Outputs, such as "development requirements for which a commitment exists" that are related to both of these Concepts are considered. Otherwise, we say that the relation is non-exclusive.

3.4 Summary

MOSAIC contains three meta-models, the IS, IC and SF Meta-Models. In this section, we introduced their elements and their relations.

The purpose of the IS and IC Meta-Model is to integrate multiple PRs. The IS Meta-Model is used to normalize the structure of the different PRs while the IC Meta-Model is used to normalize the terminology of these PRs. This integration is possible as the PracticeConcepts of the IS Meta-Model are related with the Concepts of the IC Meta-Model.

Furthermore, the IC Meta-Model together with the SF Meta-Model are used to integrate the software project context with the PRs. The purpose of the SF Meta-Model is to model the software project context. This integration is possible as the SituationalFactors of the SF Meta-Model are related with the Concepts of the IC Meta-Model.

In the following chapters, we use the abbreviation ISM, ICM and SFM in front of the elements of these models to differentiate between them and remind the reader of the model they belong to.

In the following chapter, we define guidelines how the defined meta-models can be applied and the corresponding models can be created.

4 Modeling Activities

This chapter describes the activities performed by a Modeler to create the MOSAIC models based on the MOSAIC meta-models (Fig. 14). In the following sections, we describe and exemplify the modeling activities, namely the extraction and relation of MOSAIC elements. We remind about the definition of an extraction of elements in MOSAIC.

The extraction of ISM, ICM or **SFM elements** consists of the identification of these elements in the PRs or in the situational factors framework [Clarke and O'Connor 2012], as well as their modeling according to the guidelines defined by the MOSAIC meta-models.

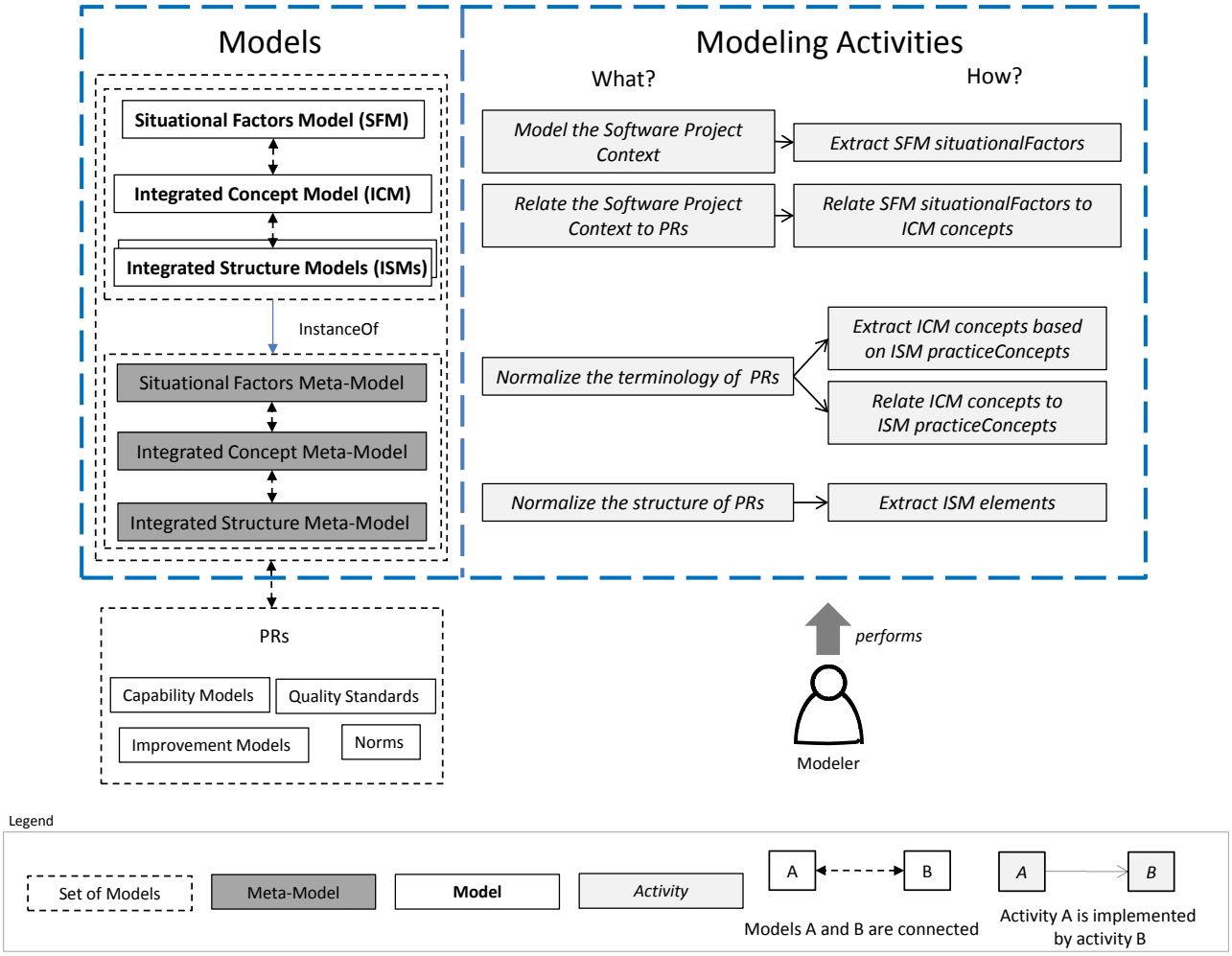


Fig. 14: Modeling activities

The modeling activities are specified using UML activity diagrams. For a better understanding of the relations between the ISM, ICM and SFM elements that are considered in the modeling activities, we also illustrate the corresponding meta-model together with each activity diagram. For simplicity reasons, the activity diagrams do not visualize the objects produced by each activity.

4.1 Running Example

After a description of the modeling activities, we illustrate the modeling of the ISM practice CMMI-DEV PPQA SP2.1 “Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers” and of the SFM situationalFactor “application quality”. We selected this ISM practice from our scenario related to the application quality as we can illustrate the modeling of the following elements:

- Various types of ISM practice elements: ISM activities, inputs, outputs and roles
- More than one ISM activity
- ISM implicit practice elements
- More than one ICM concept for one ISM practiceConcept
- Various ICM conceptCategories (inclusively ICM conceptCategory related to different status of an ICM concept)
- Different similarity relations between ICM concepts
- Relations between the SFM situationalFactor and ICM abstract concepts.

As the ICM abstract concept is an important role of an ICM concept in a generalizationOf-mono-hierarchy, we remind about its definition. We also remind about generalizationOf-mono-hierarchy.


Reminder

An ICM **abstract concept** is a ICM general concept and is the root of the generalizationOf-mono-hierarchy.
A **generalizationOf-mono-hierarchy** is an hierarchy where one ICM special concept can have only one ICM general concept.

4.2 Structure Normalization

To normalize the structure of a PR, a Modeler extracts its ISM elements. The activity diagram visualizes the extraction of ISM elements related to one ISM practice, as well as the IS Meta-Model (Fig. 15). The ISM practice language elements are not visualized because they only describe the ISM practice elements.

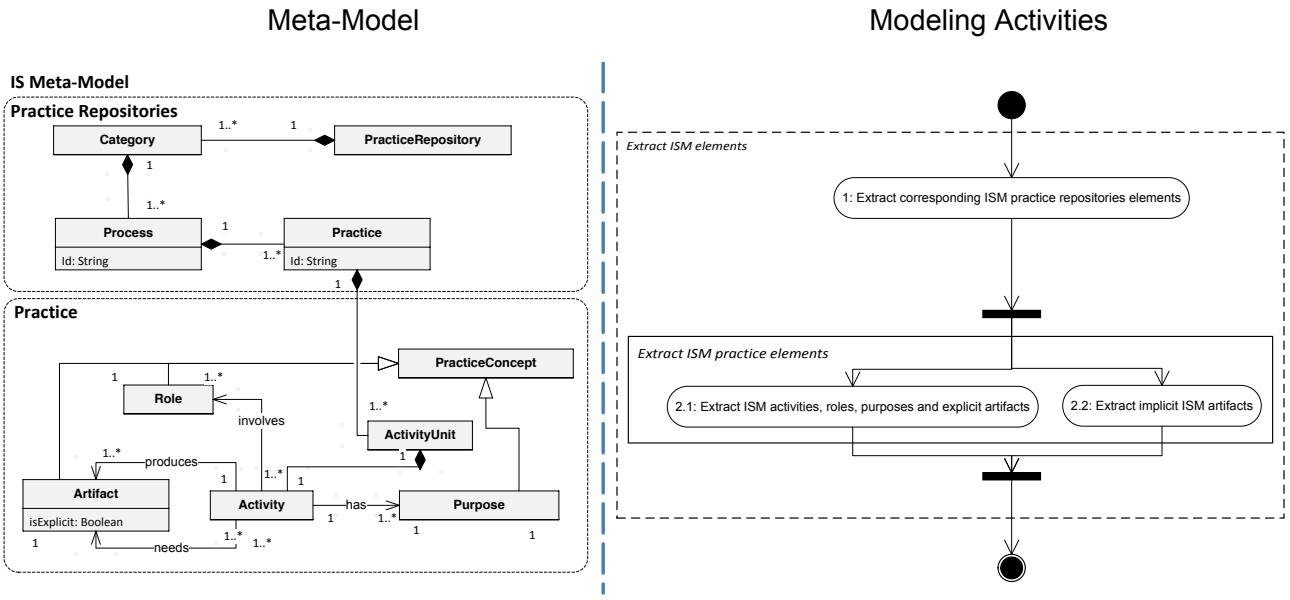


Fig. 15: IS Meta-Model and the extraction of ISM elements

4.2.1 Extraction of ISM Practice Repositories Elements

The Modeler *extracts the corresponding ISM practice repositories elements*, i.e. the ISM practiceRepository, category, process and practice based on the PR title, category title, process title and practice title corresponding to the considered ISM practice. Then, he sets the ids of the ISM process and practice based on the corresponding ids defined in the PRs.

4.2.2 Extraction of ISM Practice Elements

The Modeler *extracts the ISM practice elements*. Consequently, the Modeler *extracts the ISM activities*, *roles*, *purposes and explicit artifacts* contained in the ISM practice. He also *extracts the ISM implicit artifacts* contained in the PR' description of the ISM practice.

4.2.3 Example

We extracted the ISM elements of the ISM practice CMMI-DEV PPQA SP2.1 "Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers" (Fig. 16).

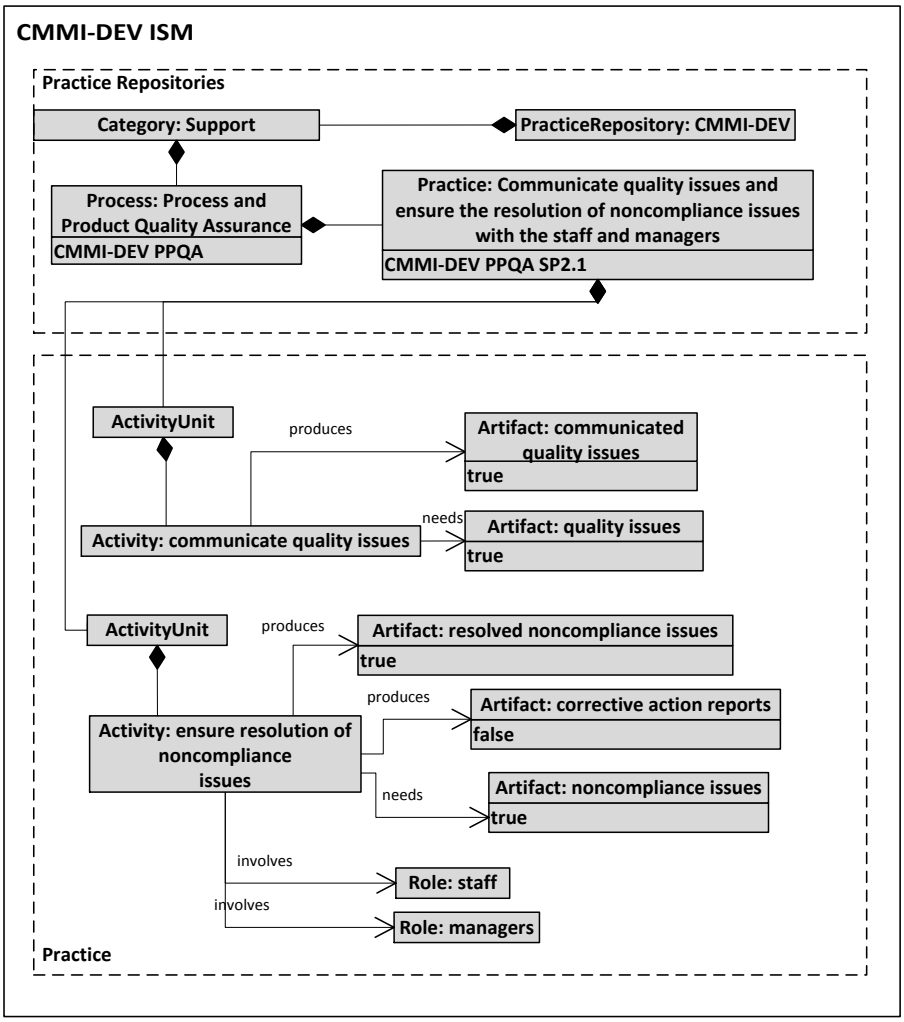


Fig. 16: Example – Structure normalization – Extraction of ISM elements

In the first step, the Modeler extracts the ISM practiceRepository, category, process and practice (act. 1). Once the ISM practice is extracted, the Modeler extracts the ISM activities, roles and explicit artifacts contained in the ISM practice (act. 2.1). Finally, the Modeler identifies the ISM implicit artifacts, such as "corrective action reports" (act. 2.2). This is listed as typical work product in the CMMI-DEV practice description.

4.3 Terminology Normalization

To normalize the terminology of a PR, the Modeler extracts ICM concepts based on ISM practiceConcepts and relates them.

There are two modalities for the extraction of concepts to create an ontology [Lembo et al. 2006]. Firstly, an existing ontology can be used by making simple modifications to fit our purposes (Ontology Customization). We used OntoGen[4] [Li Liao et al. 2005] to generate an ontology using the ISM practices and the PRs' description of these ISM practices. This did not deliver good results so that

[4] OntoGen at http://ontogen.ijs.si/

not simple, but complex modifications would have been necessary to achieve an ICM that can serve our purposes. Secondly, concepts can be extracted by starting with the most special concepts and then creating the most general ones (Bottom-Up Construction). MOSAIC is based on this strategy and thus, the Modeler extracts the ICM special concepts based on ISM practiceConcepts and then creates the most general ones, the ICM abstract concepts.

4.3.1 Extraction of ICM Concepts based on ISM PracticeConcepts

For simplicity reasons, we visualize the extraction of only one ICM concept based on an ISM practiceConcept. According to its definition, the extraction of MOSAIC elements consists of two main activities: identification and modeling of these elements. Consequently, the extraction of ICM concepts consists of the following activities:

- Identify an ICM concept in the ISM practiceConcept
- Model the ICM concept
    - Search the ICM concept
    - Create the ICM concept and its ICM abstract concept (if the ICM concept is not found in the previous step)
    - Define the relations for the ICM concept and the ICM abstract concept (if these ICM concepts are created in the previous steps)

The last three sub-activities have to be decomposed and consist of sub-activities. Fig. 17 gives an overview of all the activities needed to extract an ICM concept. These activities are individually vizualized in the following sections.

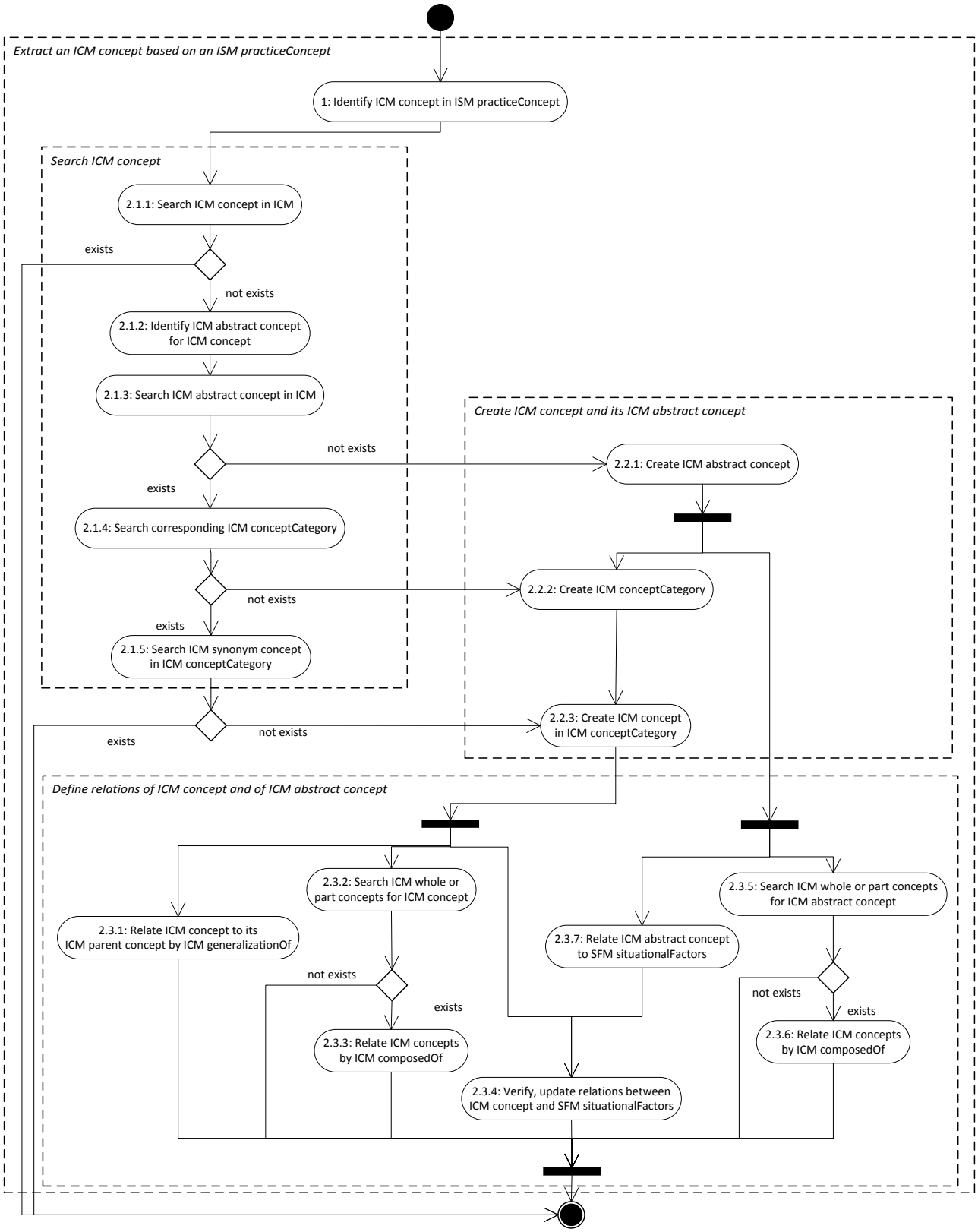


Fig. 17: Terminology normalization – Extraction of an ICM concept based on an ISM practiceConcept

In the following, we describe these activities and where necessary, define guidelines for the creation of the ICM elements. As the last three activities contain further activities, we visualize them in an activity diagram together with the IC Meta-Model.

4.3.1.1 Identification of the ICM Concept in the ISM Practice Element

The Modeler *identifies an ICM concept in the ISM practiceConcept* by analyzing this ISM practiceConcept. An ISM practiceConcept can be semantically represented by one or more ICM concepts. An ICM concept is the smallest combination of words contained in the ISM practiceConcept that has a unique meaning in the context of PRs.

4.3.1.2 Search of the ICM Concept

After the identification of the ICM concept in the ISM practiceConcept, the Modeler *searches the ICM concept* in ICM (Fig. 18). First, the Modeler *searches an ICM concept* that is syntactically equal to the identified ICM concept. If such ICM concept exists, the extraction activity finishes. If it does not exist, the Modeler tries to find an ICM synonym concept. On that account, the Modeler *identifies its ICM abstract concept for the ICM concept* and *searches the ICM abstract concept in ICM*. If this ICM abstract concept does not exist, then there is no ICM synonym concept in ICM and the ICM concept must be created. We remind about the definition of an ICM synonym concept.

Reminder

ICM synonym concepts have the same semantic meaning and thus, they are semantically equal and refer to the same ICM concept.

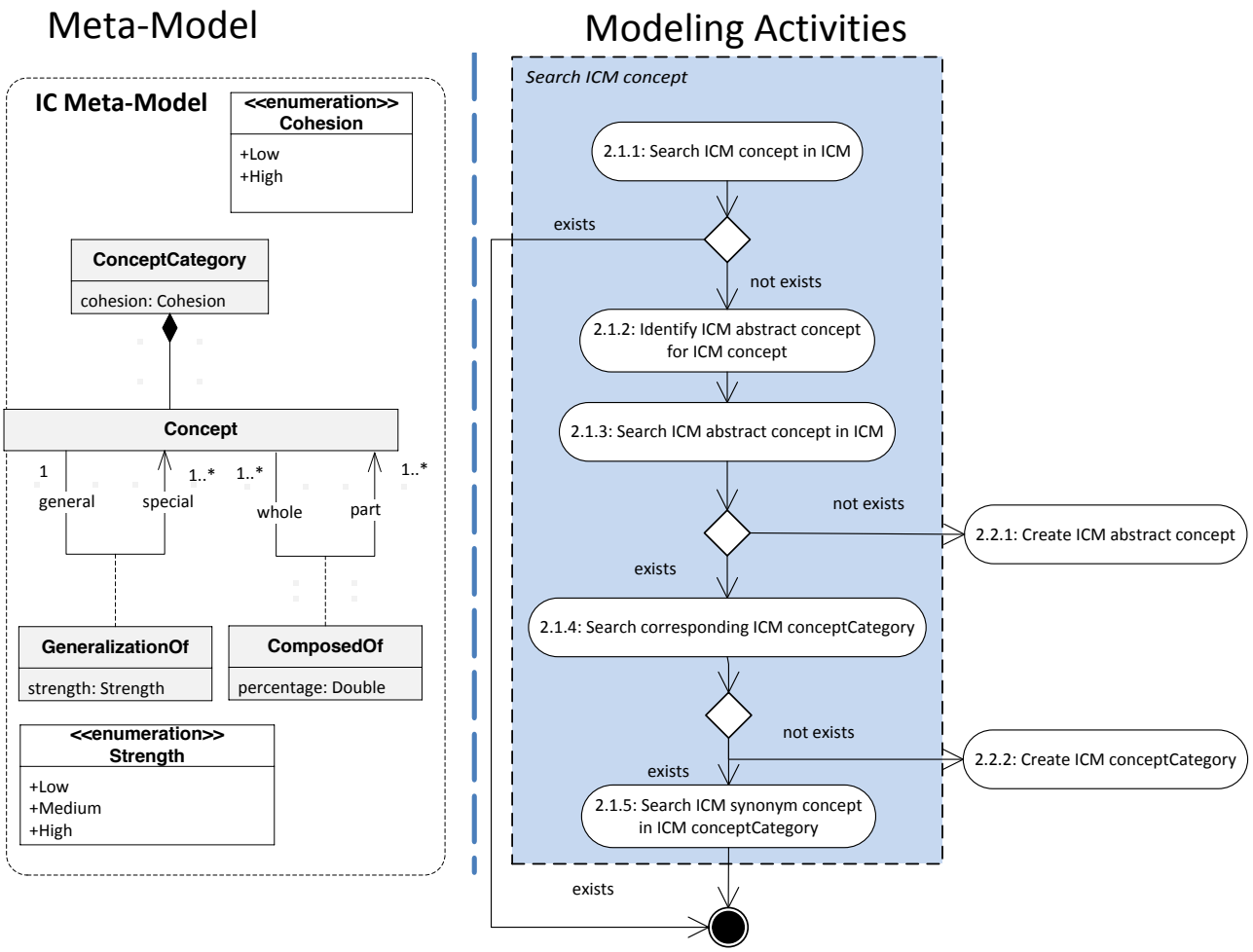


Fig. 18: IC Meta-Model and the search of an ICM concept

If the ICM abstract concept exists, the Modeler *searches the corresponding ICM conceptCategory* by traversing the generalizationOf-mono-hierarchy of the ICM abstract concept. According to the graph theory, there are two modalities to traverse a tree: depth-first or breadth-first. As the ICM conceptCategory names are not unique on the different levels, the generalizationOf-mono-hierarchy should be breadth-first searched to identify the corresponding ICM conceptCategory. Otherwise, the ICM conceptCategory name is found on a deeper level as it should and the Modeler misses the searched ICM synonym concept. If the ICM conceptCategory is not found on a certain level, the Modeler traverses the next level. The steps of traversing the generalizationOf-mono-hierarchy are visualized in Fig. 19 together with a dummy generalizationOf-mono-hierarchy.

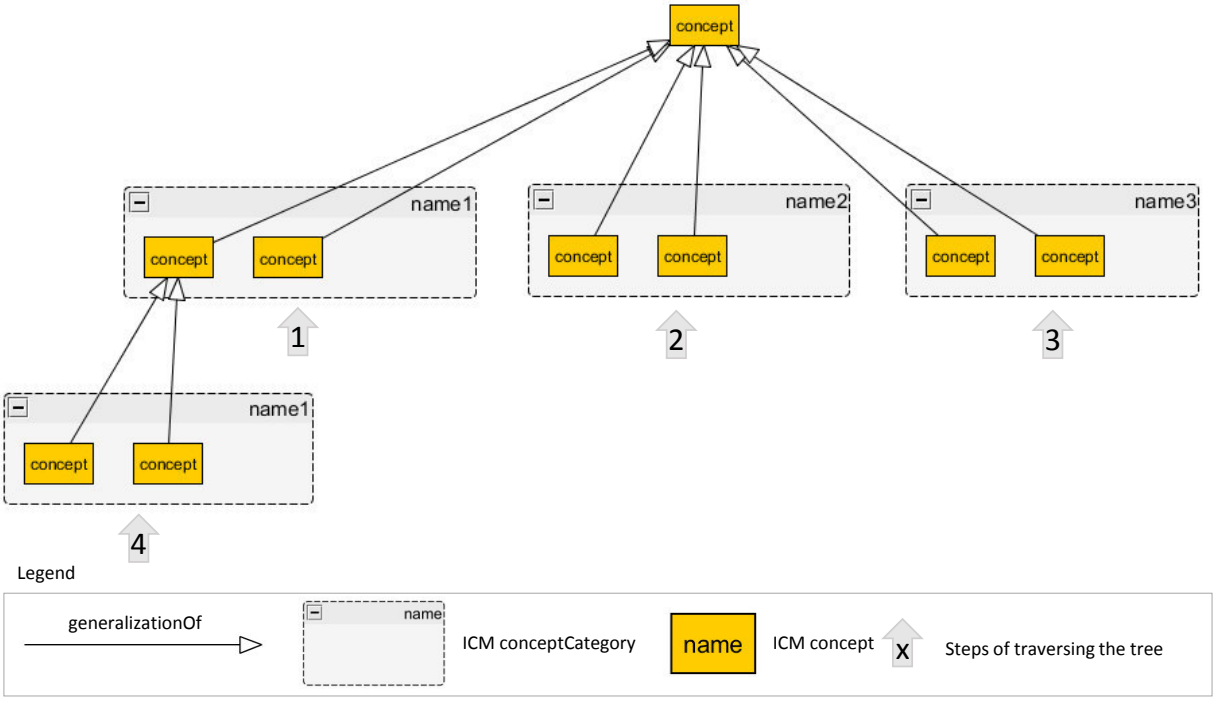


Fig. 19: Breadth-first traversing of the generalizationOf-mono-hierarchy in ICM

If the ICM conceptCategory is found, then the Modeler *searches for an ICM synonym concept* in this ICM conceptCategory. If he finds it, the search finishes, otherwise the Modeler has to *create the ICM concept*.

4.3.1.3 Creation of the ICM Concept and its ICM Abstract Concept

If the ICM concept is not found in ICM, the Modeler *creates an ICM concept and its ICM abstract concept* (Fig. 20). We recommend to create ICM concepts in their plural form as the ICM concepts can be related to ISM artifacts or roles. For example, the ICM concept “stakeholders” can refer to the ISM artifact that represents a list of stakeholders or to the ISM role that represents the person stakeholder. Consequently, the plural form avoids the modeling of unnecessary ICM concepts by creating only one of them for the two types of ISM practiceConcepts. Furthermore, based on our experiences, we argue that the plural form increases the readability of the ICM and the understanding of its ICM concepts.

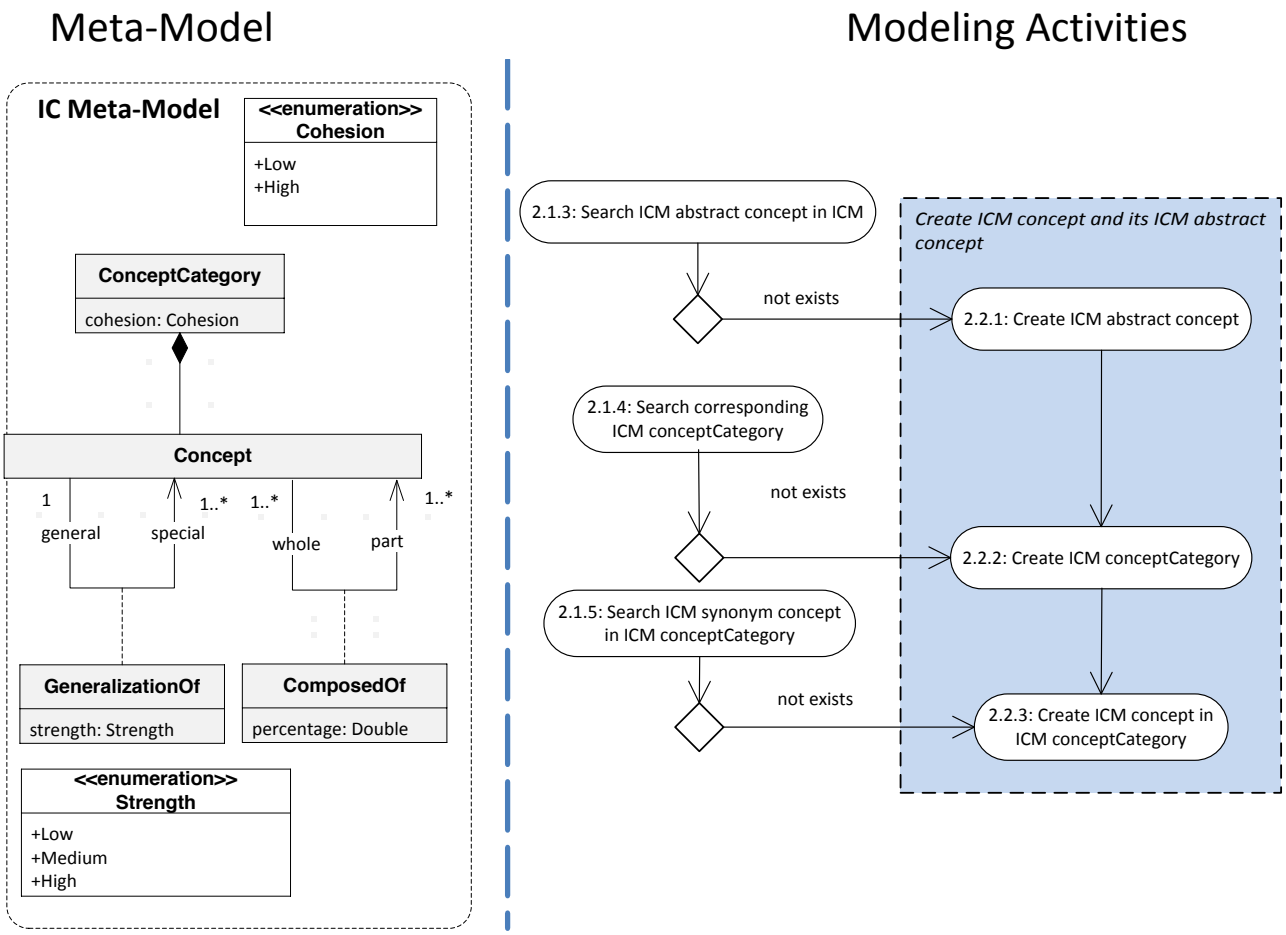


Fig. 20: IC Meta-Model and the creation of an ICM concept and its ICM abstract concept

As for each ICM concept, the ICM also contains its ICM abstract concept, the Modeler has to *create this ICM abstract concept* if it does not exist. Furthermore, the Modeler *creates the ICM conceptCategory* in the generalizationOf-mono-hierarchy. Based on our experiences with the extraction of a high number of ICM concepts, we propose the following ICM conceptCategories:

- **Context** to reflect the software area in which the ICM concepts are used (e.g. "development stakeholders", "service operation stakeholders").
- **Parties** to define who is responsible for the ICM concept in a software project (e.g. "supplier requirements", "project requirements").
- **Scope** to reflect the purpose of certain ICM concepts (e.g. "categorization risk parameters", "analysis risk parameters").
- **Type** to reflect the different ICM concept types (e.g. "requirements", "designs" are different types of "work products").
- **Status** to express the state of an ICM concept that changes when different activities are performed ("maintained project plans"). Inspired by the Deming-Cycle (Plan-Do-Check-Act) and by an approach that uses these values to mark the outcomes of the CMMI-DEV practices [Chen and Staples 2007], we differentiate between various "Status" types to reflect the different states of an ICM concept:
    - **Status**: ICM concept is created (e.g. "Establish a work breakdown structure").

- **Status-Do**: ICM concept is implemented (e.g. “Perform the project plan”).
- **Status-Check**: ICM concept is verified (e.g. “Analyze the software requirements”).
- **Status-Do-Check**: ICM concept is verified and according to this verification it is updated (e.g. “Maintain project plan”).

Each of the defined status is valid for the ICM concepts extracted from PRs and not from other sources. For example, the term “maintain” used in the description of the function points analysis method does not only refer to the verification and update activity, but also to the adding, changing or deleting activity (Longstreet, 2012).

Furthermore, the creation of an ICM conceptCategory also involves the definition of its cohesion (“High” or “Low”) to reflect the similarity degree of ICM concepts within an ICM conceptCategory.

The last step is the *creation of the ICM concept* in the ICM conceptCategory.

4.3.1.4 Definition of Relations of the ICM Concept and of its ICM Abstract Concept

Once the ICM concept is created, the Modeler *defines the relations of the ICM concept and of its ICM abstract concept* to ICM similar concepts or to SFM situationalFactors (Fig. 21). For a better readability, we do not illustrate the IC Meta-Model.

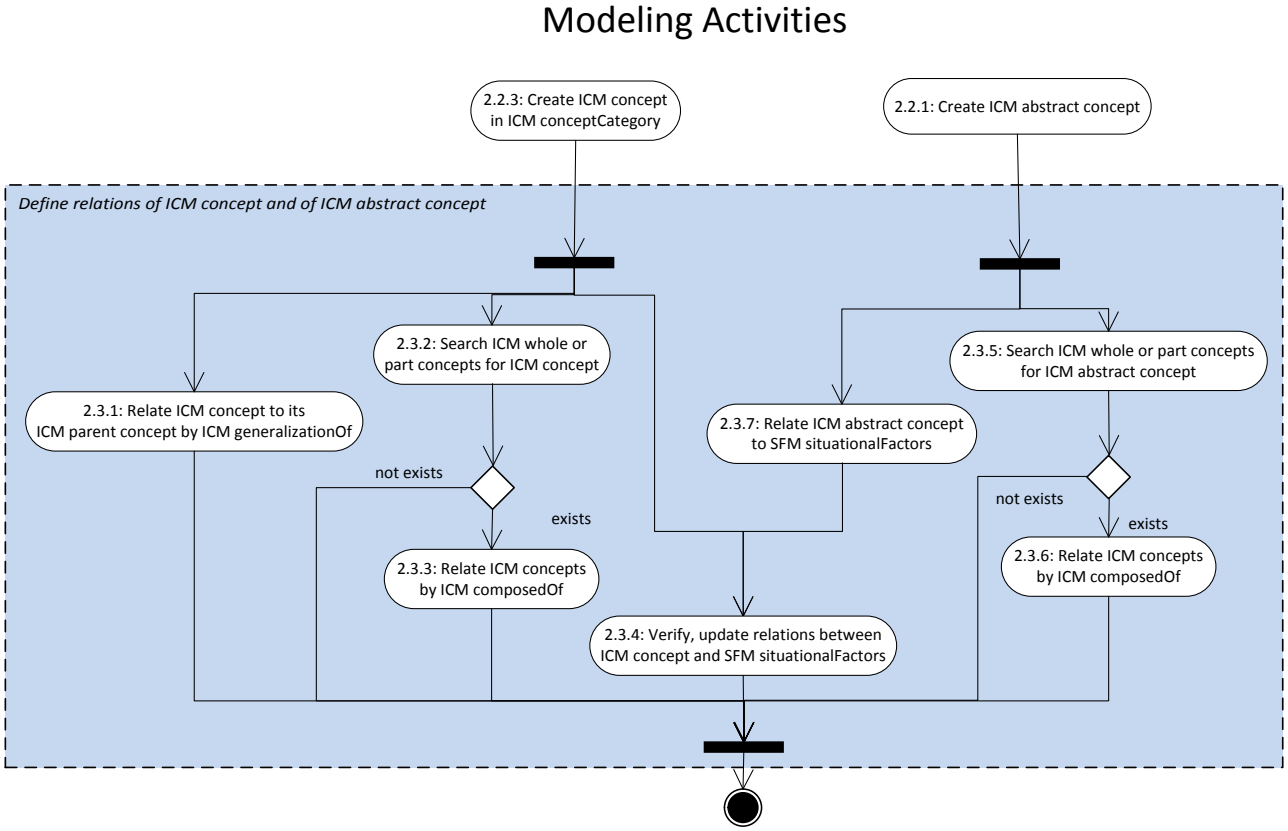
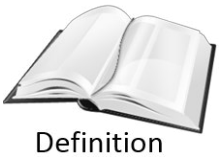


Fig. 21: Relation of an ICM concept and its ICM abstract concept

To identify the similarity relations between the ICM concepts, we analyzed ontologies or schema-based matching approaches that might be applied to support the identification of similar elements.


Definition

Schema-based matching is the process of identification of two elements that are semantically related.

Many diverse solutions for schema-based matching have been proposed so far [Shvaiko et al. 2005; Rahm and Bernstein 2001]. These propose to compare the terms in a schema or ontology by using linguistic resources, such as lexicons, thesauri, graph matching algorithms or structure metadata. We evaluated some online dictionary tools, such as WordNet[5], Rensselaer MSR Server[6] and Wikipedia Miner[7] that use these linguistic resources. Unfortunately, their ability to identify ICM similar concepts are not satisfactory. The reasons can be that the dictionaries mostly contain general terms and not the specific PRs' terminology or that the similarity relations between the specific terms are not consistently documented. Therefore, the Modeler has to manually relate the ICM concepts.

The relation of ICM concepts leads to creation of hierarchies in ICM. This relation is performed as follows.

Firstly, the Modeler *relates the ICM concept to its ICM parent concept by the ICM generalizationOf*. He sets the strength for this relation, i.e. he selects one of the values "Low", "Medium" or "High". Based on the ideas from ontology construction and on our experience extracting ICM concepts, we define some guidelines to select one of these three values. In the ontology construction, there is a differentiation between "self-standing" concepts (e.g. "stakeholders", "programmers") and "value-types" concepts (e.g. "key stakeholders") [Rector 2002]. While the "value-types" concepts characterize their parent concept by using an adjective, the "self-standing" concepts are nouns. Therefore, we propose the following strengths:

- **"High"** to relate an ICM parent concept and a "value-types" ICM concept in all ICM conceptCategories.
- **"Medium"** to relate an ICM parent concept and a "self-standing" ICM concept in the ICM conceptCategory **Context**.
- **"Low"** to relate an ICM parent concept and a "self-standing" ICM concept in the ICM conceptCategories **Status**, **Parties**, **Scope** and **Type**.


Remark

The strength for an ICM generalizationOf cannot be automatically set based on the afore-mentioned guidelines. It has to be individually defined by the Modeler depending on the similarity between the ICM concepts as there can be situations where our guidelines do not apply.

[5] WordNet at http://wordnet.princeton.edu/
[6] Rensselaer MSR Server at http: //cwl-projects.cogsci.rpi.edu/msr/
[7] Wikipedia Miner at http://wdm.cs.waikato.ac.nz:8080/

Secondly, the Modeler relates the created ICM concept to ICM whole or part concepts. He *searches for the ICM whole or part concepts* and then *relates the found ICM concepts by the ICM composedOf*. This is an exception as we restrict the definition of the ICM composedOf between ICM descendant concepts. Furthermore, the Modeler has also to set the percentage for this relation, i.e. he defines a rational value between 0 and 1 meaning the percentage one ICM concept covers the other ICM concept.

Thirdly, the Modeler relates the created ICM concept to SFM situationalFactors. He *verifies and updates the relations between the ICM concept and SFM situationalFactors*.

The Modeler verifies if such relations are possible for a SFM situationalFactor. If this is true, then the Modeler verifies if these relations already exist in ICM between this SFM situationalFactor and the ICM abstract concept. If this is valid, no activity has to be performed. If one of these relations does not exist for the ICM abstract concept, then this relation between the ICM concept and the SFM situationalFactor is created.

The Modeler also have to verify, if the existing relations between the SFM situationalFactor and the ICM abstract concept are also valid for the current ICM concept. If there exist relations that are not valid, then these relations have to be removed and be defined for all ICM concepts except the current ICM concept. Consequently, the relations between the ICM concepts in the generalizationOf-mono-hierarchy and the SFM situationalFactor have to be updated.

If the ICM abstract concept is created for the ICM concept, then the Modeler has to analogously relate this ICM abstract concept with the ICM whole or part concepts. Furthermore, the Modeler relates the ICM abstract concept to SFM situationalFactors (section 4.4.2).

4.3.2 Relation of ICM Concepts to ISM PracticeConcepts

For simplicity reasons, we visualized the relation of only one ICM concept to an ISM practiceConcept (Fig. 22). First, the Modeler *searches the corresponding ICM concept in ICM*. If this ICM concept is found, the Modeler relates it to the ISM practiceConcept. Otherwise, the activity finishes and the extraction of this ICM concept based on the ISM practice element must be performed first (see previous section).

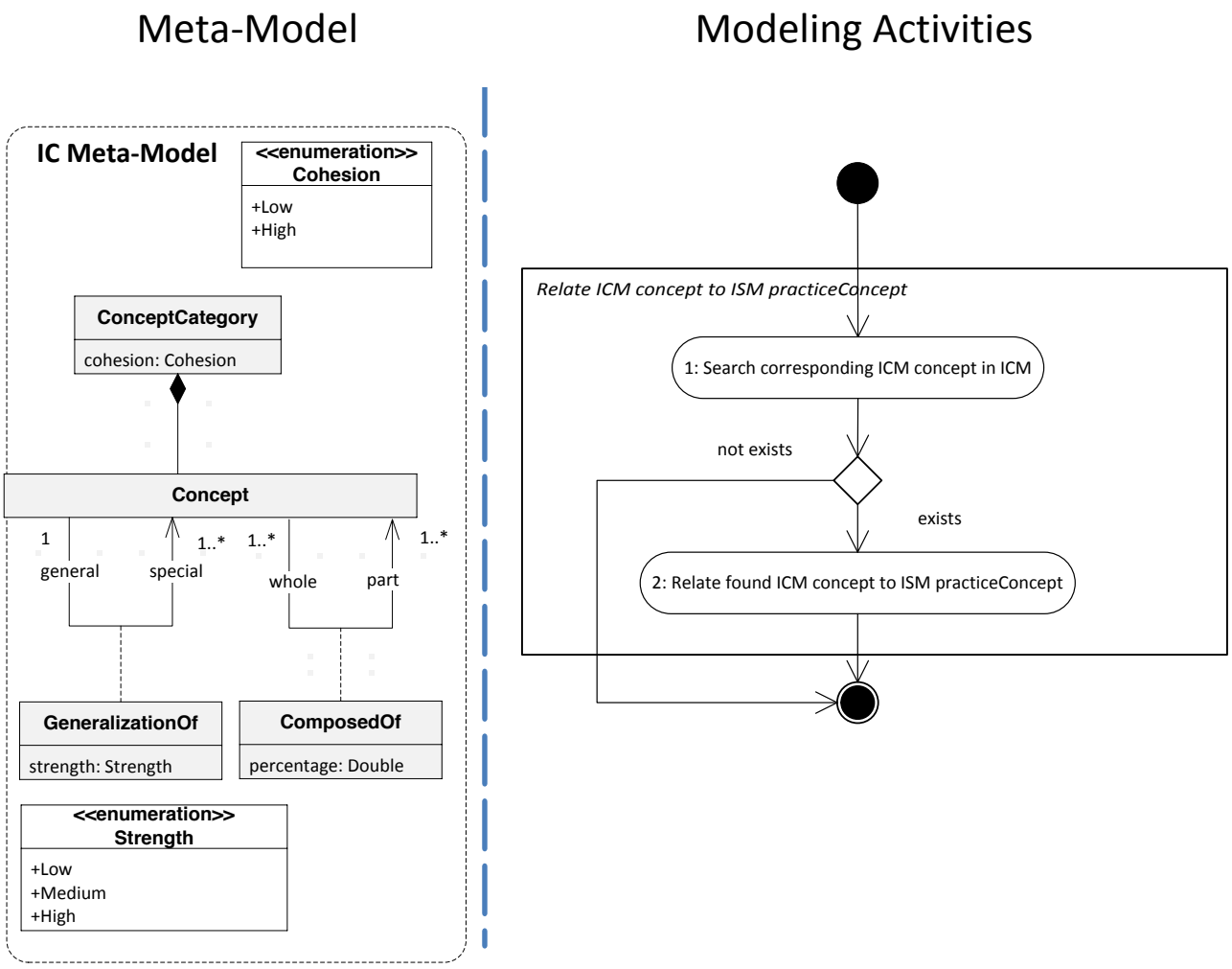


Fig. 22: IC Meta-Model and the relation of an ICM concept to an ISM practiceConcept

4.3.3 Example

We extracted the ICM concepts for the ISM practiceConcepts of the ISM practice CMMI-DEV PPQA SP2.1 "Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers". Fig. 23 illustrates the corresponding ISM and ICM after the extraction of the ICM concepts and their relation to the ISM practiceConcepts. For simplicity reasons, we do not illustrate the ICM concepts of the "corrective action reports".

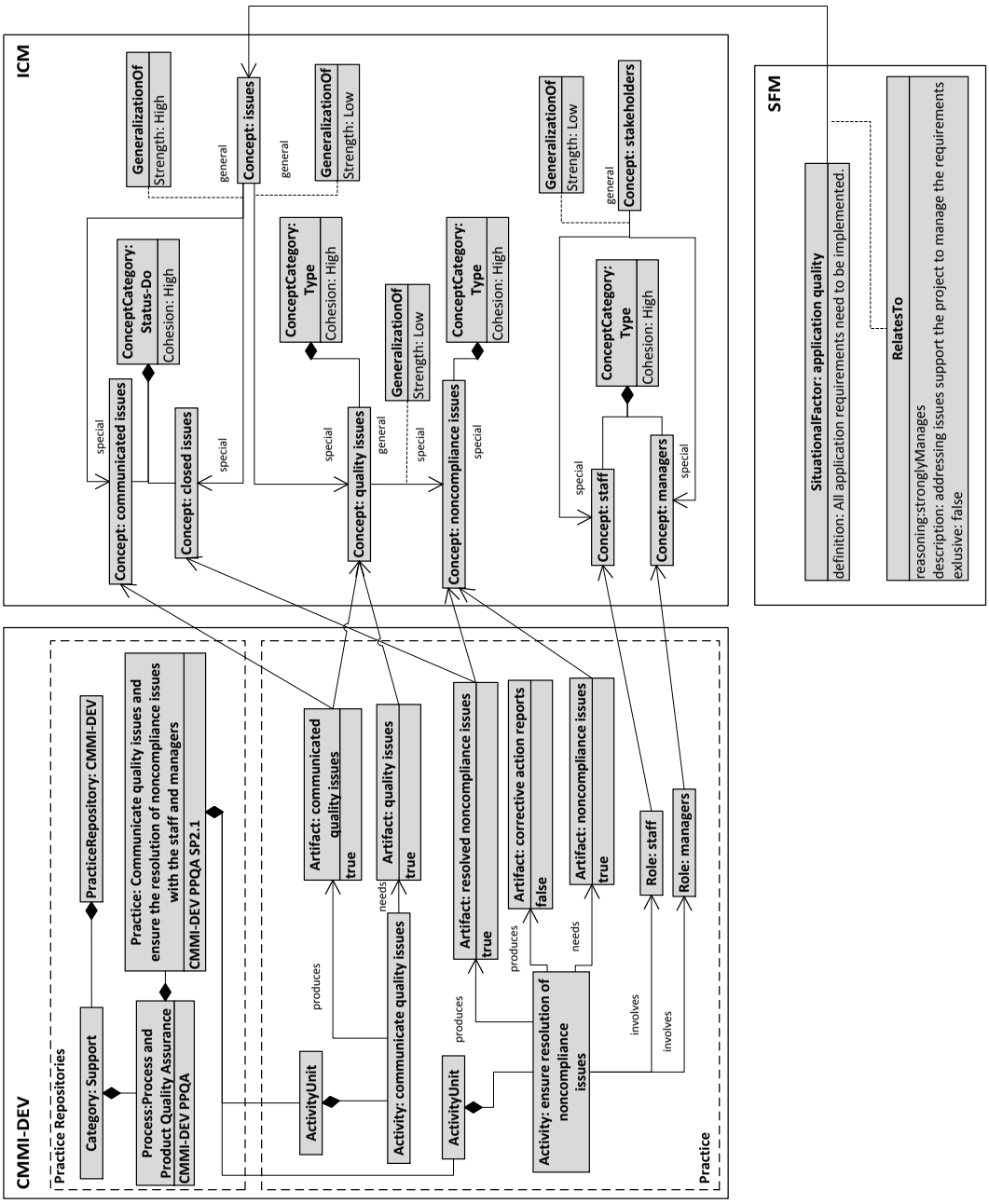


Fig. 23: Example – Extraction of ICM concepts and their relation to ISM practiceConcepts

For a better understanding, we describe the extraction of ICM concepts and their relation to the ISM practiceConcept “resolved noncompliance issues”.

We suppose that the ISM, ICM and SFM have the following configuration (Fig. 24).

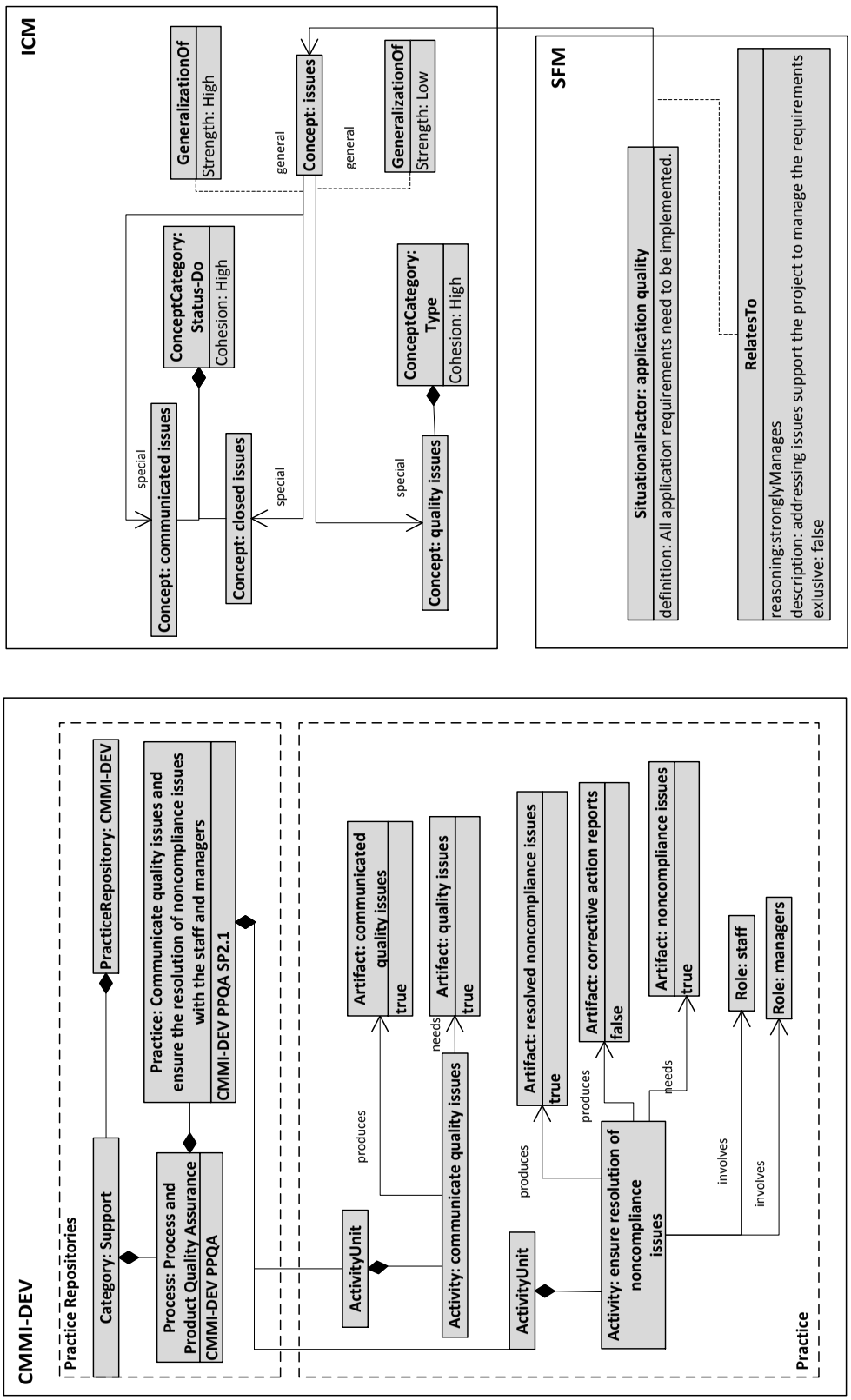


Fig. 24: Example – ISM, ICM and SFM before the extraction of ICM concepts and relation to ISM practiceConcepts

The extraction of ICM concepts based on the ISM practiceConcept “resolved noncompliance issues” is performed in two iterations:

- Extraction of the ICM concept “noncompliance issues”
- Extraction of the ICM concept “resolved issues”

In the first iteration, the Modeler extracts the ICM concept “noncompliance issues” and performs the following activities:

- He identifies the ICM concept “noncompliance issues” (act. 1).
- He searches for this ICM concept in the ICM:
    - He searches for an ICM concept syntactically equal to “noncompliance issues” (act. 2.1.1).
    - As he does not find it, he determines the ICM abstract concept “issues” (act. 2.1.2).
    - He searches it in the ICM (act. 2.1.3).
    - He traverses the generalizationOf-mono-hierarchy of the ICM abstract concept “issues” and searches for the corresponding ICM conceptCategory (act. 2.1.4) until the ICM parent concept “quality issues” is found. Here, he does not find the corresponding ICM conceptCategory “Type” and thus, the ICM concept does not exist in the ICM.
- As the ICM abstract concept exists, the Modeler only creates the ICM concept:
    - He defines the ICM conceptCategory “Type” to reflect the different “quality issue” types (act. 2.2.2).
    - He adds the ICM concept “noncompliance issues” to the ICM conceptCategory (act. 2.2.3).
- Once the ICM concept is created, the Modeler relates it to ICM similar concepts and SFM situationalFactors.
    - He relates the ICM concept “noncompliance issues” to its ICM parent concept “quality issues” by the ICM generalizationOf. For this relation, he defines the strength “Low” as the ICM conceptCategory is “Type” (act. 2.3.1).
    - He searches for ICM whole or part concepts (act. 2.3.2). Such ICM concepts are not found.
    - He verifies the relations of its ICM abstract concept with the SFM situationalFactors (act. 2.3.5). The relation between the SFM situationalFactor and the ICM abstract concept is also valid for the ICM concept because the adoption of the ICM concept “noncompliance issues” strongly manages a situation in the software project described by the “application quality”. The “noncompliance issues” needs to be addressed by such software projects.

In the second iteration, the Modeler extracts the ICM concept “resolved issues” and performs the following activities:

- He identifies the ICM concept “noncompliance issues” (act. 1).
- He searches for the ICM concept:
    - He searches for an ICM concept that is syntactically equal (act. 2.1.1). He does not find it.
    - As he does not find it, he determines the ICM abstract concept “issues” (act. 2.1.2).
    - He searches it in the ICM (act. 2.1.3).
    - He traverses the generalizationOf-mono-hierarchy of the ICM abstract concept “issues” to search for the corresponding ICM conceptCategory (act. 2.1.4). He finds the ICM conceptCategory “Status-Do”.

- He searches for an ICM synonym concept in this ICM conceptCategory. He finds the ICM concept “closed issues”. Consequently, no activity has to be performed any more.

Finally, the Modeler connects the two extracted ICM concepts to the ISM practiceConcept. Therefore, the ISM practiceConcept “resolved noncompliance issues” is related to the ICM concepts “noncompliance issues” and “closed issues”.

4.4 Integration of the Software Project Context

The Modeler models the software project context and relates it to the PRs by extracting the SFM situationalFactors from the situational factors framework [Clarke and O’Connor 2012] and relating these SFM situationalFactors to ICM concepts.

For simplicity reasons, we describe the extraction of only one SFM situationalFactor and its relation to ICM concepts.

4.4.1 Extraction of SFM SituationalFactor

The Modeler *extracts the SFM situationalFactor* from the situational factors framework. As this framework does not provide a definition of the situational factors, the Modeler defines the SFM situationalFactor based on the list of sub-factors contained in the framework.

4.4.2 Relate SFM SituationalFactor to ICM Concepts

Fig. 25 illustrates the relation of one SFM situationalFactor to an ICM abstract concept or its ICM descendant concepts in a generalizationOf-mono-hierarchy. The figure contains also the SF Meta-Model and the ICM element *Concept* for a better understanding of the elements considered in this modeling activity.

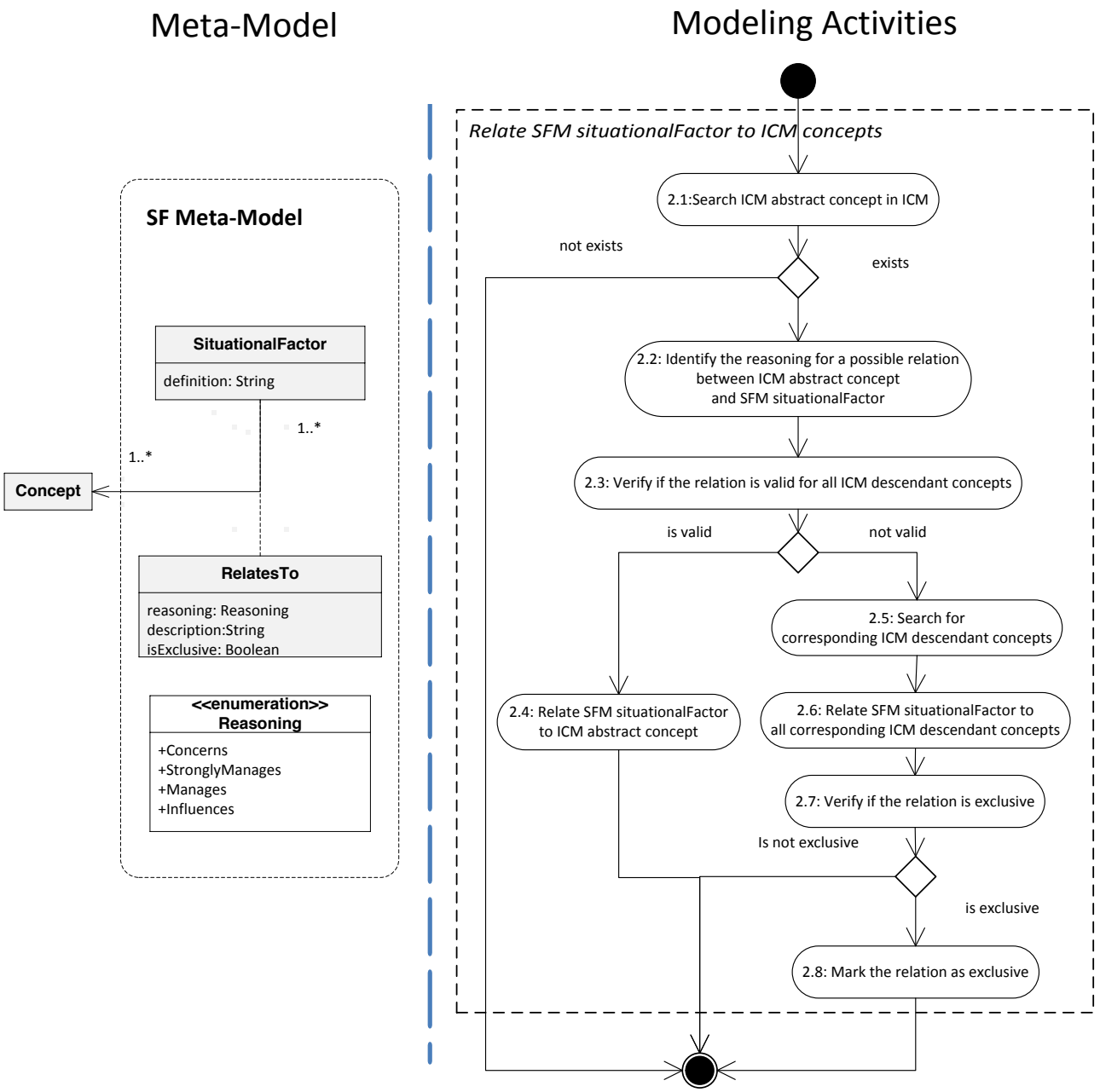


Fig. 25: SF Meta-Model and the relation between a SFM situationalFactor and ICM concepts

To relate a SFM situationalFactor to one or more ICM concepts, the Modeler *searches for an ICM abstract concept in ICM*.

If the Modeler finds the ICM abstract concept, he *identifies the reasoning for a possible relation between the ICM abstract concept and SFM situationalFactor*. Then, the Modeler traverses the whole generalizationOf-mono-hierarchy of this ICM abstract concept to *verify if this relation is valid for all its ICM descendant concepts*. If the relation is valid, the Modeler *relates the SFM situationalFactor with the ICM abstract concept*. For this relation, he specifies the values for its attributes, such as the reasoning and description. The relation between a SFM situationalFactor and an ICM abstract concept

is always non-exclusive as this relation refers to only one ICM concept. This attribute is per default false and thus, the Modeler does not have to define it.

If the relation is not valid for all its ICM descendant concepts, the Modeler *searches for the ICM descendant concepts* for which the relation is valid. He performs a depth-first search, i.e. it traverses the tree until he finds the ICM parent concept whose all ICM descendant concepts can be related to the SFM situationalFactor. When these ICM concepts are found, the Modeler *relates the SFM situationalFactor to all these corresponding ICM descendant concepts*. He also specifies the values for the relation's attributes, such as the reasoning, description and isExclusive.

4.4.3 Example

We extracted the SFM situationalFactor "application quality" and related it to the ICM concepts corresponding to the ISM practice CMMI-DEV PPQA SPI2.1 "Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers".

Fig. 26 illustrates the ICM and SFM after the relation of the SFM situationalFactor to ICM concepts. We supposed that the ICM contains the two ICM abstract concepts ("issues" and "stakeholders") and their ICM descendant concepts as illustrated in Fig. 26.

The Modeler extracts the SFM situationalFactor "application quality" (act. 1). Based on its subfactors listed in the situational factors framework ("required application quality", "maintainability"), he specifies the definition of the SFM situationalFactor.

Then, the Modeler searches for ICM concepts that can be related to the SFM situationalFactor and performs the following activities:

- He searches for an ICM abstract concept in ICM (act. 2.1) and identifies the ICM abstract concept "issues" that can be related with the SFM situationalFactor.
- He determines the reasoning "Manages" for this relation (act. 2.2) as the "application quality" is managed by the adoption of the ICM abstract concept "issues".
- He verifies if this relation is valid for all the ICM descendant concepts of the ICM abstract concept "issues" (act. 2.3). The relation is valid for all of its ICM concepts.
- As this relation is valid, the Modeler relates the SFM situationalFactor to this ICM abstract concept (act. 2.4). This activity finishes.

The relation between the SFM situationalFactor and the ICM abstract concept "stakeholders" is defined analogously.

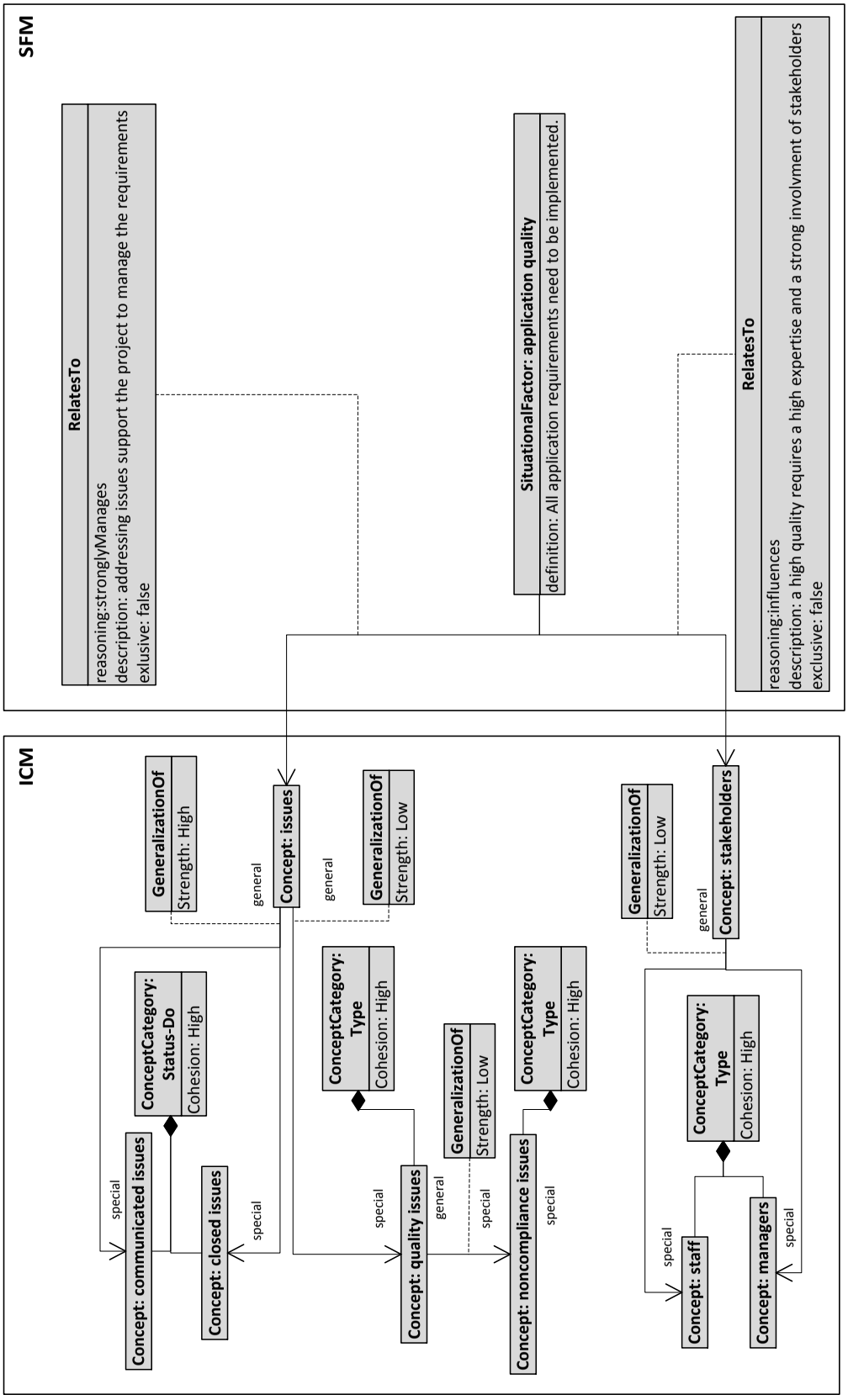


Fig. 26: Example - Modeling of a SFM situationalFactor and its relation to ICM concepts

4.5 Summary

A Modeler can perform various modeling activities to create the MOSAIC models based on the MOSAIC meta-models. Therefore, he receives guidelines to create the ISMs for each PR, the ICM, the SFM and the relations between them. As the ICM is the most important model of MOSAIC, we aim to particularly support a Modeler in its creation. First, we tried to generate the ICM based on the description of ISM practices so that the Modeler has to perform only simple modifications. This generation did not deliver promising results. Moreover, we tried to generate the similarity relations between the ICM concepts. However, the existing schema-based matching resources could not be used. Therefore, we defined modeling activities to support a Modeler to create step by step the ICM and provided him additional guidelines for its creation.

For a better understanding, we described these modeling activities in detail, defined their sequences and exemplified them for an ISM practice from our example scenario related to the application quality.

5 Analysis Activities and Metrics

Based on the MOSAIC models, various activities can be performed by an Analyzer to gain the needed information from multiple PRs. These activities are realized by different algorithms which are based on metrics. In the following, we start with a short introduction of the measurement theory. Next, for each analysis activity we define the associated metrics and algorithms.

5.1 Running Example

In the following sections, we illustrate the analysis activities on the following ISM practices within our example scenario related to the application quality:

- CMMI-DEV PPQA SP1.2.5 “Identify each case of noncompliance found during evaluations.”
- CMMI-DEV PPQA SP2.1 “Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers.”
- SPICE SUP.1 BP9 “Ensure resolution on non-conformances.”

We selected these ISM practices as they support us in illustrating the following:

- Selection of the ISM practices CMMI-DEV PPQA SP2.1 and SPICE SUP.1 BP9 based on the relation between the SFM situationalFactor and their common ICM abstract concept
- Identification of similar ISM practices based on the relations between ICM concepts and their ISM practiceConcepts:
  - Identification of a “High” similarity degree between the CMMI-DEV PPQA SP2.1 and SPICE SUP.1 BP9 practices
  - Identification of a coverage degree of 1 of the CMMI-DEV PPQA SP2.1 considering the SPICE SUP.1 BP9, i.e. the CMMI-DEV practice covers the SPICE practice
  - Identification of the output state “Do” of the CMMI-DEV PPQA SP2.1 and SPICE SUP.1 BP9
- Identification of a dependency degree “Strong” between the CMMI-DEV practices based on the relation between ICM concepts and their ISM practiceConcepts

5.2 Measurement Theory

According to Fenton and Pfleeger, measurement is “*the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to describe them accordingly to clearly defined rules*” [Fenton and Pfleeger 1997]. Metrics specify this assignment and refers to “*a defined measurement method and the measurement scale*” [ISO/IEC 14598-1:1999 1999]. Measures refer to a “*variable to which this value is assigned*” [ISO/IEC 25000:2014 2014] and not to the method itself nor the scale. Metrics, as measurement methods, are part of MOSAIC and thus, are in the focus of this chapter.

Rules, such as assumptions and requirements have to be defined to guarantee the validity of the metrics. Assumptions are needed to allow the assignment of numbers or symbols to the attributes of entities. Based on these assumptions, the developed metrics have to fulfill requirements.

Furthermore, a scale must be defined to perform this measurement. There are four types of a measurement scale to assign numbers or symbols to attributes of entities:

- *Nominal scale*. The attributes of entities are classified using different symbols (e.g. red, brown or blonde for the hair color of a person)
- *Ordinal scale*. The attributes of entities are classified and ordered, so that operations, such as "greater than", "less than" or "equal to" can be performed. However the classifications do not provide information about the distance between the entities in this order (e.g. small, medium or high for the height of a person).
- *Interval scale*. The attributes are classified, ordered and the distance between them is equal. Several operations are allowed with the exception of the ratio operation (e.g. values for the Celsius temperature).
- *Ratio scale*. This scale has all the properties of the interval scale and in addition possesses a meaningful (unique and non-arbitrary) zero value. For this reason, the ratio operation is allowed on this scale (e.g. values for the length of a train).
- *Absolute scale*: It is a ratio scale, where the result value is not only proportional to the searched one, but it represents this value itself. It is valid for natural values, that are resulted by counting.

5.3 MOSAIC Metrics

In this section, we give an overview of the MOSAIC metrics and describe how we developed them. Table 9 gives an overview of the metrics defined in MOSAIC to implement the analysis activities and thus, to achieve our goals.

Metrics		Final Result	Analysis activities / MosAIC main goals
Support metrics		Support degree of ISM practices for a softare project whose context is described by a certain SFM situationalFactor	Selection of ISM practices based on SFM situationalFactors
Practice Similarity Metrics	Similarity metrics	Similarity degree of two or more ISM practices	Identification of similar ISM practices
	Coverage metrics	Coverage degree of ISM practices	
	Output state metrics	Output states of ISM practices	
Dependency metrics		Dependency degree between ISM practices	Identification of dependencies between ISM practices

Table 9: Overview of the MOSAIC metric types

All these metrics are defined based on the results obtained by a first development of the similarity metrics. The similarity metrics are first defined by a student and by us during a bachelor thesis at our research department [Pyatkova 2011], evaluated and continuously calibrated in various iterations [Jeners et al. 2012b; Jeners et al. 2012c; Jeners and Lichter 2013].

According to the measurement theory, we have to define assumptions, requirements and a scale for each metric type: support, similarity, coverage, output state and dependency metrics.

Therefore, for each metric type, we define specific assumptions and requirements. One generic requirement is valid for all metrics:

R-All. The metric should be differentiable, comparable, reproducible and plausible [Ludewig and Lichter 2010]. These terms mean in detail:

- *Differentiable*. Different inputs for a metric cause different results.
- *Comparable*. The results are comparable. This is possible for results on a ordinal or ratio scale.
- *Reproducible*. The same input always leads to the same value.
- *Plausible*. The results reflect the human experts' opinion, i.e. are aligned with the human experts' results.

Furthermore, each metric type is based on a different measurement scale and thus, we specify it when we give more details about each metric type.

For each metric type we apply the following design principle (Fig. 27).

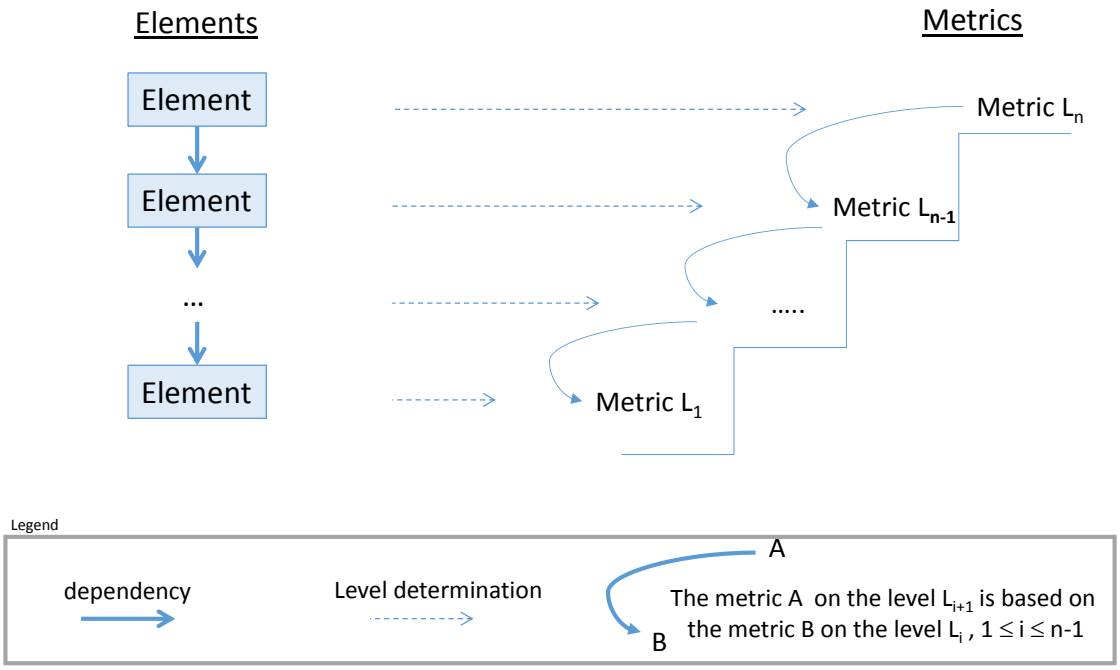


Fig. 27: MOSAIC metrics – Design principle

The MOSAIC metrics are defined on different levels (1 – n) that are determined by elements that are related by semantic dependencies. These elements and their dependencies are given by the elements and their relations in the IS and IC Meta-Models.

For example, three levels are given by ISM activityUnits that contain ISM practiceConcepts that are related to ICM concepts. Based on the metric for ICM concepts (level L_1), we define the metric for ISM practiceConcepts (level L_2) and then, the metric for ISM activityUnit (level L_3).

Consequently, for each level i ($1 \leq i \leq n-1$ and n is the total number of levels), a metric is defined using the following design principles:

- Each metric on the level L_{i+1} depends on the metric on the previous level L_i.
- The metric result on the level L_{i+1} is high[8] on the given scale if the metric result on the previous level L_i is also high[9] on the given scale for this level.
- The metric result on the level L_n represents the final result.

[8] High is defined for each metric type in the sections 5.4.1, 5.5.2.1, 5.5.3.1, 5.5.4.1, 5.6.1

[9] See previous footnote

The MOSAIC metrics are also SMART. A set of criteria is proposed to assess the metrics on the basis of the 5W+H rule (What, Who, Where, When, Why + How) [Abran and Buglione 2008]. The MOSAIC metrics fulfill these criteria:

- ***S**pecific (What)*: Each metric has a purpose (Table 9).
- ***M**easurable (How)*: Each metric is based on the similarity relations between the ICM concepts and the semantic dependencies between the ISM elements. It is defined on different levels. For each level, thresholds for the given scale for each metric are given.
- ***A**dd Value & Actionable (Why)*: The metrics support the analysis activities in organizations that work with multiple PRs.
- ***R**elevant (Who)*: An Analyzer uses the analysis activities that are based on these metrics.
- ***T**ime (When)*: The analysis activities and thus, the metrics can be used when two or more PRs need to be compared, when practices from multiple PRs need to be selected or when the dependencies between PRs need to be identified.

In the following, we describe for each analysis activity the algorithms and underlying metrics that are the basis for each activity.

We describe the MOSAIC metrics using various functions and sets of elements of the MOSAIC meta-models. We do not completely formalize the description of these metrics for a better readability and understanding. This formalization is possible and is performed within the implementation of the algorithms and their underlying metrics by the MOSAIC Toolbox.

Before going into details, we introduce the following terms to denote the MOSAIC elements that are used as parameters by these algorithms and metrics:

- $pract_{ISM}$ for an ISM practice
- au_{ISM} for an ISM activityUnit
- pc_{ISM_t} for an ISM practiceConcept of type t, $t \in Type = \{ISM\ output, ISM\ input, ISM\ role, ISM\ purpose\}$.
- $conc_{ICM}$ for an ICM concept
- sf_{SFM} for a SFM situationalFactor

5.4 Selection of ISM Practices

In this section we describe how MOSAIC implements the selection of ISM practices based on SFM situationalFactors. We propose the support metrics and an algorithm that can be used to automatically identify best suited ISM practices based on the software project context.

For the selection of ISM practices, we consider the mappings between ISMs, ICM and SFM. Furthermore, we consider ISM practiceConcepts pc_{ISM_t} that are ISM explicit artifacts of type $t \in T$, $T = \{ISM\ input, ISM\ output\}$. We consider only ISM inputs and outputs because the adoption of ISM roles and purposes do not explicitly support a software project, but allow that the ISM artifacts are adopted to achieve a purpose. Furthermore, we only consider ISM explicit artifacts because the ISM

implicit artifacts are only examples of outputs or inputs specified in the PRs' description for an ISM practice.

5.4.1 Support Metrics

We define support metrics to determine the support degree of an ISM practice for a software project whose context is described by a certain SFM situationalFactor. Based on this support degree, ISM practices can be selected to support a software project.

According to the measurement theory, we define the following specific assumption (A1) and requirement (R1) for the support metrics:

A1. ISM outputs are more important than ISM inputs regarding their support degree for a project.
R1. The computed support degree reflects the importance of ISM practiceConcepts.

The assumption (A1) specifies that the support degree of ISM outputs is stronger than the support degree of ISM inputs. This is because the ISM outputs reflect the practice actual work, which is expected to be performed. The ISM input only contributes to the creation of the ISM output. Consequently, the adoption of the ISM outputs is more important than the adoption of the ISM inputs to support the software project in a critical situation. The requirement (R1) requires that this importance has to be reflected by the metric result.

According to the measurement theory, we have to define the measurement scale for the support metrics. The support metrics use an ordinal scale with the order: "Strong" > "Medium" > "Absent".

According to our design principle, we define the support metrics on different levels (Fig. 28).

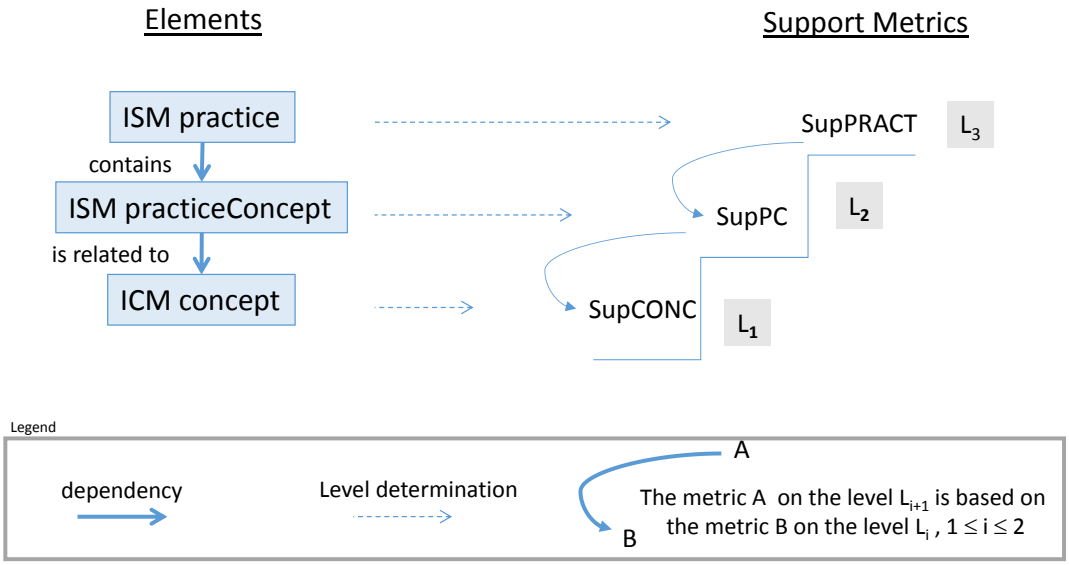


Fig. 28: Support metrics – Design

Firstly, each support metric depends on the support metric on the previous level. The support metric on the ISM practice level depends on the support metric on the ISM practiceConcept level that depends on the support metric on the ICM concept level.

Secondly, the support degree on a level is high if the support degree on the previous level is also high. For example, the support degree of an ISM practice is "Strong", if the support degree of one of

its ISM practiceConcepts is “Strong”. Analogously, the support degree of the ISM practiceConcepts is “Strong”, if the support degree of corresponding ICM concepts is “Strong”.

Finally, the support degree of the ISM practices represents the final result.

5.4.1.1 Support Metrics for ICM Concepts

For a $conc_{ICM}$, the metric **SupCONC (Support of Concepts)** determines its support degree for a software project whose context is described by the sf_{SFM}.

SupCONC depends on the reasoning of the relation between the SFM situationalFactor sf_{SFM} and the ICM concept $conc_{ICM}$.

The SupCONC value is “Strong”, if the reasoning has the value “Concerns” or “StronglyManages”. This means that the adoption of the ICM concept concerns or strongly manages a situation in the software project and therefore, it is strongly recommended. The SupCONC value is “Medium”, if the reasoning has the value “Manages” or “Influences”. This means that the adoption of the ICM concept manages a situation in the project or a situation in the project can influence the adoption of the ICM concept and thus, the project members have to pay attention about the impact of this situation on the project. Consequently, the adoption of the ICM concept is relevant for the software project. Otherwise, the SupCONC value is “Absent”. This means that the ICM concept does not have to be adopted.

The metric is defined as follows:

$$SupCONC(sf_{SFM}, conc_{ICM}) = \begin{cases} \textit{„Strong“}, & \text{iff there is a relation between } sf_{SFM} \text{ and } conc_{ICM} \text{ with the reasoning „Concerns“ or „StronglyManages“.} \\ \textit{„Medium“}, & \text{iff there is a relation between } sf_{SFM} \text{ and } conc_{ICM} \text{ with the reasoning „Manages“ or „Influences“.} \\ \textit{„Absent“}, & \text{otherwise.} \end{cases}$$

5.4.1.2 Support Metrics for ISM PracticeConcepts

For a pc_{ISM_t} of type $t \in T$, the metric **SupPC (Support of Practice Concepts)** determines its support degree for a software project whose context is described by the sf_{SFM}.

Let $CONC_{pcISM\_t} = \{conc_{ICM} \mid conc_{ICM} \text{ is related to } pc_{ISM\_t}\}$. This set contains all ICM concepts related to pc_{ISM_t}.

SupPC depends on the metric SupCONC for the ICM concepts related to pc_{ISM_t}. First, the SupPC value is “Strong”, if and only if pc_{ISM_t} of type ISM output, otherwise it is “Medium” or “Absent”. Furthermore, the SupPC value is given by the corresponding SupCONC values of the related ICM concepts. The SupPC value can be “Strong” or “Medium”, if the SupCONC values for all related ICM concepts are “Strong”. The SupPC value can be “Medium”, if it does not exist a SupCONC value for a related ICM concept that is “Absent”, i.e. the SupCONC values are “Strong” or “Medium”. Otherwise, the SupPC value is “Absent”.

The metric is defined as follows:

$SupPC(sf_{SFM}, pc_{ISM\_t}) =$

- ***„Strong“***, iff $\forall conc_{ICM} \in CONC_{pcISM\_t}$, $SupCONC(sf_{SFM}, conc_{ICM})$ = „Strong“ and t=ISM output.
- ***„Medium“***, iff one of the two conditions holds:
  - $\forall conc_{ICM} \in CONC_{pcISM\_t}$, $SupCONC(sf_{SFM}, conc_{ICM})$ = „Strong“ and t=ISM input.
  - $\nexists\ conc_{ICM} \in CONC_{pcISM\_t}$, $SupCONC(sf_{SFM}, conc_{ICM})$ = „Absent“ and t=ISM output.
- ***„Absent“***, otherwise.

5.4.1.3 Support Metrics for ISM Practices

For a $pract_{ISM}$, the metric **SupPRACT (Support of Practices)** determines its support degree for a software project whose context is described by sf_{SFM}.

Let $PC_{practISM\_t} = \{pc_{ISM\_t} \mid pc_{ISM\_t}$ is contained in $pract_{ISM}\}$. This set contains all ISM practiceConcepts contained in $pract_{ISM}$.

SupPRACT depends on the metric SupPC for the ISM practiceConcepts contained in $pract_{ISM}$. The SupPRACT value is given by the corresponding SupPC values of the contained ISM practiceConcepts. The SupPRACT value is “Strong”, if there exists one ISM practiceConcept for which its SupPC value is “Strong”. Otherwise, SupPRACT value is “Medium”, if there exists one ISM practiceConcept for which its SupPC value is “Medium”. Otherwise, SupPRACT value is “Absent”.

$SupPRACT(sf_{SFM}, pract_{ISM}) =$

- ***„Strong“***, iff $\exists pc_{ISM\_t} \in PC_{practISM\_t}$, $SupPC(sf_{SFM}, pc_{ISM\_t})$ = „Strong“.
- ***„Medium“,*** iff both conditions hold:
  - $\nexists\ pc_{ISM\_t} \in PC_{practISM\_t}$, $SupPC(sf_{SFM}, pc_{ISM\_t})$ = „Strong“.
  - $\exists pc_{ISM\_t} \in PC_{practISM\_t}$, $SupPC(sf_{SFM}, pc_{ISM\_t})$ = „Medium“.
- ***„Absent“***, otherwise.

An ISM practice with the support degree „Strong“ or „Medium“ for a SFM situationalFactor is called “Strong” or „Medium“ ISM practice respectively.

5.4.1.4 Requirements Verification

We verify the achievement of the requirements that the support metrics need to fulfill.

First, the general requirement for all metrics (R-All) requests that the metrics’ results have to be differentiable, comparable, reproducible and plausible. As the support metrics are based on the rela-

tion between SFM situationalFactors and ICM concepts and its attributes, the results are differentiable, comparable and reproducible. The plausibility of the results are verified during several experiments performed with several experts (section 8.3).

Furthermore, the specific requirement (R1) requires that the importance of the ISM practiceConcepts has to be considered. According to the assumption (A1), an ISM output is more important than an ISM input. This is considered as the SupPRACT value for an ISM practice can only be "Strong" if this ISM practice contains an ISM output for which the SupPC value is "Strong". If the ISM practice contains ISM inputs for which the SupPC value is "Strong" or "Medium", then the SupPRACT value is only "Medium".

5.4.2 Algorithm

The algorithm identifies the support degree of ISM practices based on a SFM situationalFactor. It receives as input a sf_{SFM} and delivers as output selected ISM practices for a software project whose context is described by this sf_{SFM}. Therefore, it delivers the ISM practices $pract_{ISM}$ with their corresponding SupPRACT values "Strong" and "Medium".

As the algorithm is based on the relations between the SFM situationalFactor and ICM concepts, and this kind of relation can be exclusive or not, we remind about its definition.


Reminder

The relation between an ICM concept and a SFM situationalFactor r_{SFM} is **exclusive**, when exactly this combination of related ICM concepts have to be adopted, i.e. only the ISM inputs and outputs that are related to all these ICM concepts have to be adopted. Otherwise, we say that r_{SFM} is non-exclusive.

The algorithm consists of the following steps:

1. Let SUP_PRACT = {}.
2. For each relation rSFM, identify CONC ={concICM | concICM is related to sfSFM by the relation rSFM}.
3. If the relation rSFM is non-exclusive, identify CONCD = {concICM | concICM is an ICM descendant concept of concICM ∈ CONC}. Let CONC = CONC ∪ CONCD.
4. If the relation rSFM is non-exclusive, identify PCt = {pcISM_t | pcISM_t is related to one or more concICM ∈ CONC, ∀ t ∈ T}. Otherwise, identify PCt = {pcISM_t | pcISM_t is related to all concICM ∈ CONC ∀ t ∈ T }.
5. For each pcISM_t ∈ PCt do:
   5.1. Identify the PRACT ={practISM | practISM contains pcISM, ∀ t ∈ T }.

5.2. For each practISM ∈ PRACT:

5.2.1. Identify PCpractISM_t = {pcISM_t | pcISM_t is contained in practISM}.

5.2.2. For each pcpractISM_t ∈ PCpractISM_t do:

5.2.2.1. Identify CONCpcISM_t = {concICM | concICM is related to pcpractISM_t}.

5.2.2.2. For each concICM ∈ CONCpcISM_t compute the SupCONC(sfSFM,concICM).

5.2.2.3. Compute the SupPC(sfSFM, pcpractISM_t) based on the computed SupCONC values.

5.2.3. Compute SupPRACT(sfSFM, practISM) based on the computed SupPC values.

5.2.4. If SUP_PRACT(sfSFM,practISM) ≠ "Absent" then SUP_PRACT = SUP_PRACT ∪ {(practISM, SupPRACT(sfSFM,practISM)) }.

6. Return SUP_PRACT.

The set SUP_PRACT contains "Strong" or "Medium" ISM practices that can be selected for the software project whose context is described by sf_{SFM}.

5.4.3 Example

We illustrate the selection of the following ISM practices based on the SFM situationalFactor "application quality":

- CMMI-DEV PPQA SP2.1 "Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers."
- SPICE SUP.1 BP9 "Ensure resolution on non-conformances.

The selection is based on the relations between the SFM situationalFactors, ICM concepts and ISM practiceConcepts. Fig. 29 illustrates the modeling of the corresponding ISM practices, their related ISM elements and of the SFM situationalFactor. The illustrated ISMs, ICM and SFM contain only the needed elements to exemplify the selection.

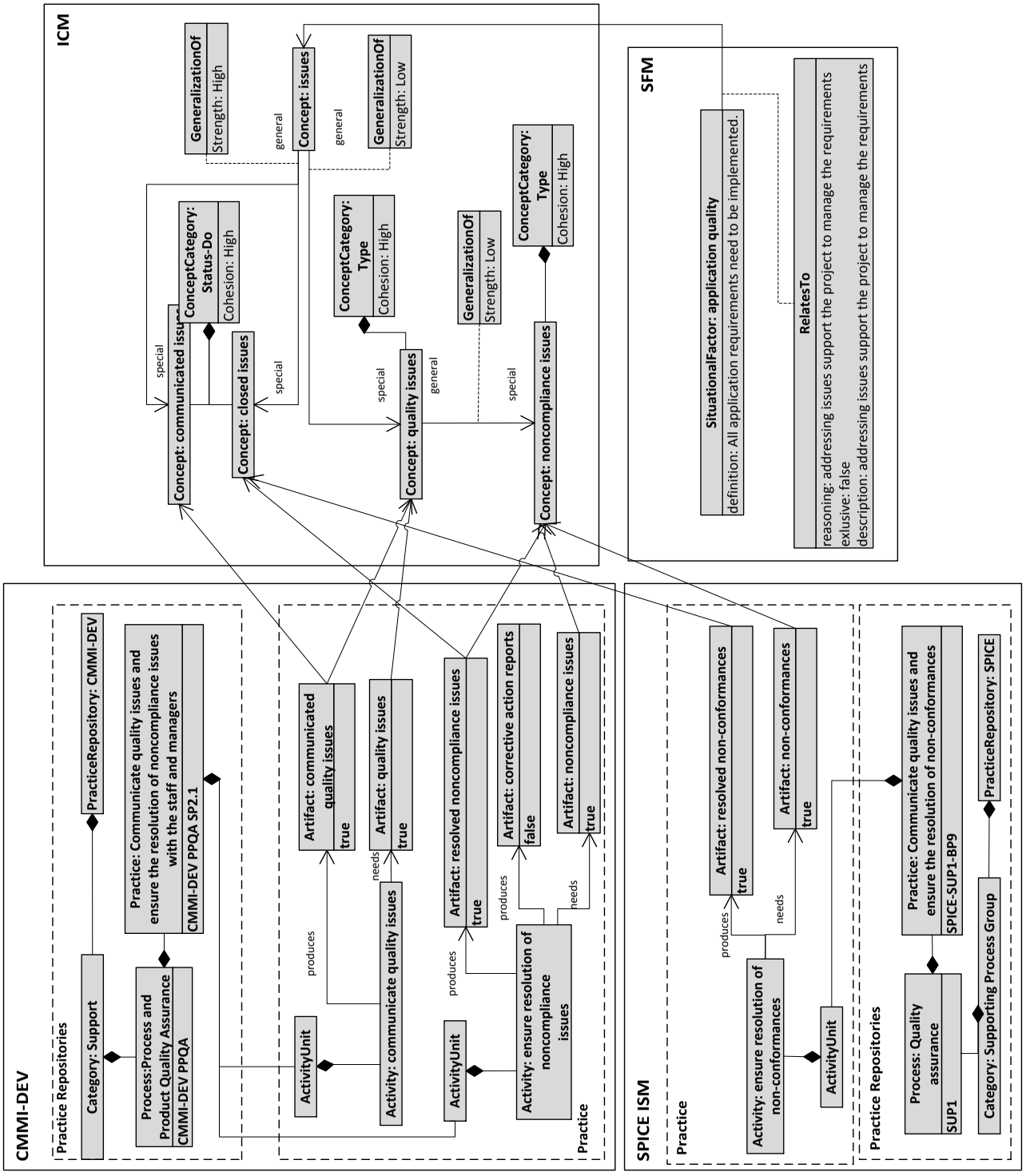


Fig. 29: Example – Selection of ISM Practices – Excerpt of ISMs, ICM and SFM

Table 10 illustrates the steps performed to compute the support degree of ISM practices for a SFM situationalFactor. This example also illustrates that the importance of the ISM practiceConcepts is relevant for the selection. For example, although the SupCONC value for the ICM concept "quality issues" is "Strong", the SupPC value is only "Medium" as the corresponding ISM practiceConcept is an ISM input.

Input	Step 1	Step 2	Step 3	Step 4
SFM sit. Factor (sf_{SFM})	SUP_PRACT	ICM concepts (CONC)	ICM concepts incl. ICM descendant concepts (CONC = CONC U $CONC_D$)	ISM practiceConcepts (PC_t)
application quality	{}	{issues}	{issues, communicated issues, closed issues, quality issues, noncompliance issues}	{communicated quality issues, resolved noncompliance issues, resolved non-conformances, quality issues, noncompliance issues, non-conformances}

Step 5							
Step 5.1	Step 5.2						
	Step 5.2.1		Step 5.2.2			Step 5.2.3	Step 5.2.4 (Output)
			Step 5.2.2.1	Step 5.2.2.2	Step 5.2.2.3		
ISM Practices in PRACT	ISM practiceConcepts in $PC_{practISM\_t}$	Type	ICM concepts in $CONC_{pcISM\_t}$	SupCONC	SupPC	SupPRACT	SUP_PRACT
CMMI-DEV PPQA SP2.1	communicated quality	ISM output	communicated issues	Strong	Strong	Strong	{(CMMI-DEV PPQA SP2.1, Strong), (SPICE SUP.1.BP9, Strong)}
			quality issues	Strong			
	resolved noncompliance		resolved issues	Strong	Strong		
			noncompliance issues	Strong			
	quality issues	ISM input	quality issues	Strong	Medium		
	noncompliance issues		noncompliance issues	Strong	Medium		
SPICE SUP.1.BP9	non-conformances	ISM input	noncompliance issues	Strong	Medium	Strong	
	resolved non-conformances	ISM output	resolved issues	Strong	Strong		
			noncompliance issues	Strong			

Table 10: Example – Selection Algorithm

5.5 Identification of Similar ISM Practices

In this section we describe how MOSAIC implements the identification of similar ISM practices. We propose practice similarity metrics and several algorithms that can be used to automatically identify similar ISM practices.

For the identification of similar ISM practices, we consider the mappings between ISMs and ICM. Furthermore, we consider ISM practiceConcepts pc_{ISM_t} that are ISM explicit artifacts. The set of ISM practiceConcepts types T is different for the three types of practice similarity metrics and algorithms. Therefore, for each practice similarity metric, we define it individually.

To define the practice similarity metrics, we first perform an analysis of the similarity theory to select the methods which can be used to identify similar elements.

5.5.1 Similarity Theory

In general, similarity is an important property because it is fundamental for human cognition. Similarity plays a key role in problem solving, remembering, prediction, and categorization [Goldstone and Son 2005]. In fact, if there were no similar objects and events, an individual would perceive

each situation as a new one and would have to learn how to use each particular object. The notion of similarity is applied in different domains. For instance, in geometry two objects are similar if they have the same shape; in psychology they are similar if they can be put into the same category. As there is no common definition of "similarity" we refer to the definition of Goodman: "*Objects are similar if they have a set of common feature*s" [Goodman 1972].

There are several methods to determine the similarity between objects based on their common features. These can be categorized as follows:

- *Spatial methods* consider objects as points or vectors in the n-dimensional space [Groenen and Borg 2013]. Well-known examples of spatial methods are the cosine distance or the euclidean distance methods**.**
- *Feature-based methods* consider objects as a finite unsorted set of features. They calculate the similarity with respect to their features. For example, the numbers of equal and non-equal features of different objects are combined to calculate their similarity [Tversky and Gati 1978].
- *Transformational methods*, e.g. the levenshtein distance [Levenshtein 1966], consider the features of two objects and their order. They count the transformations needed to convert one object into the other; i.e., the smaller the number of transformations, the higher their similarity.
- *Alignment methods*, such as structure mapping engine method [Goldstone 1994], use features of objects and their relations to determine their similarity.

As in MOSAIC the order of elements (objects' features) does not have to be considered, we do not use the transformational methods. Furthermore, the alignment methods compare two objects with features only connected by one relation. As there are different similarity relations between the ICM concepts, we do not use these methods too. We also do not use the feature-based methods as these consider only equal and non-equal but not similar objects' features.

The spatial methods consider the similarity between the objects' features by calculating their distance. For this reason, we use two spatial methods for the definition of the MOSAIC metrics: a variant of the cosine distance and the weighted euclidean distance.

First, a variant of the cosine distance method is proposed to consider the distance between features in a mono-hierarchy [Ganesan et al. 2003]. As ICM concepts are organized in generalizationOf-mono-hierarchies, we can apply this method. The similarity between features p and q of a hierarchy considers their lowest common ancestor, LCA (p, q), and the depth of p and q in the hierarchy. The similarity of p and q is high if these are located deeply in the hierarchy and their lowest common ancestor is located close to both of them.

$$\boldsymbol{Sim}\,(\boldsymbol{p},\boldsymbol{q}) = \frac{\mathbf{2}\,\boldsymbol{depth}\,(\boldsymbol{LCA}(\boldsymbol{p},\boldsymbol{q}))}{\boldsymbol{depth}(\boldsymbol{p}) + \boldsymbol{depth}(\boldsymbol{q})}$$

Second, we use the weighted euclidean distance method. This defines the similarity of two objects composed of one or more features. Each feature pair has a certain weight that defines the contribution of this pair to the final result. In MOSAIC, the ISM practices contain different ISM practiceConcepts (ISM inputs, outputs, roles or purposes) for which we define different importance values. Consequently, we can apply this method. The weighted euclidean distance between two vectors of features v_1 and v_2 of size n with their corresponding weight vector w of size n is defined as:

$$d_{v_1,v_2} = \sqrt{\sum_{i=1}^{n} w_i \, (v_{1i} - v_{2i})^2}$$

5.5.2 Identification of the Similarity between ISM Practices

According to Goodman [Goodman 1972], "*Objects are similar if they have a set of common features*". Therefore, ISM practices are similar if they have a set of common elements that are similar.

For the identification of the similarity between ISM practices, we consider ISM practiceConcepts pc_{ISM_t} of type $t \in T$, T = {ISM input, ISM output, ISM role, ISM purpose}.

5.5.2.1 Similarity Metrics

We define similarity metrics to determine the similarity degree between two or more ISM practices. According to the measurement theory, we define specific assumptions (A1 – 3) and requirements (R1 – 2) for these similarity metrics:

A1. ISM roles and purposes are the less important ISM practiceConcept types.
A2. ISM outputs are more important than ISM roles, inputs and purposes.
A3. ICM part concepts of an ICM whole concept are not similar.
R1. The computed similarity metrics' results should reflect the importance of the ISM practiceConcepts.
R2. The number of the ISM practiceConcepts of an ISM activityUnit should not influence the similarity metrics' results.

The first two assumptions (A1) and (A2) specify the importance of the ISM practiceConcepts in an ISM practice. The ISM output is more important than an ISM input, role or purpose and an ISM input is more important than an ISM role or purpose. This is because, the ISM outputs reflect the practice's actual work, which is expected to be performed. The ISM inputs contribute to the creation of the ISM outputs while the ISM roles and purposes only give additional information about how to perform an activity to produce an ISM output.

Based on these assumptions, we define the importance of the different types $t \in T$ of ISM practiceConcepts. These values are first defined based on the experience of the author and then they are continuously calibrated based on the evaluations of the similarity metrics. The ISM output has the highest value as it is the most important ISM practiceConcept. The ISM purpose has the lowest value.

Importance IMP_t	Value
IMPoutput	3.50
IMPinput	1.25
IMProle	1.00
IMPpurpose	0.25

Table 11: Importance of ISM practiceConcepts for the similarity metrics

The last assumption (A3) reflects the real world as part concepts that form a whole concept are semantically different (e.g. "wheel", "door", "engine", "seat" are parts of a "car" and are not similar).

According to the measurement theory, we also define the measurement scale for these similarity metrics. The similarity degree is given by an ordinal and a ratio scale. We use an ordinal scale with the order: “Equal” > “High” > “Medium” > “Low” > “Non-Equal” and a ratio scale with values between 0 and 1, where 1 is the highest value and means that the elements are semantically equal, 0 is the lowest value and means that the elements are semantically different.

Furthermore, all similarity metrics are symmetric. For example, for two ISM practices, the SimPRACT ($pract_{ISM1}$, $pract_{ISM2}$) = SimPRACT ($pract_{ISM2}$, $pract_{ISM1}$).

According to our design principle, we define the similarity metrics on different levels (Fig. 30).

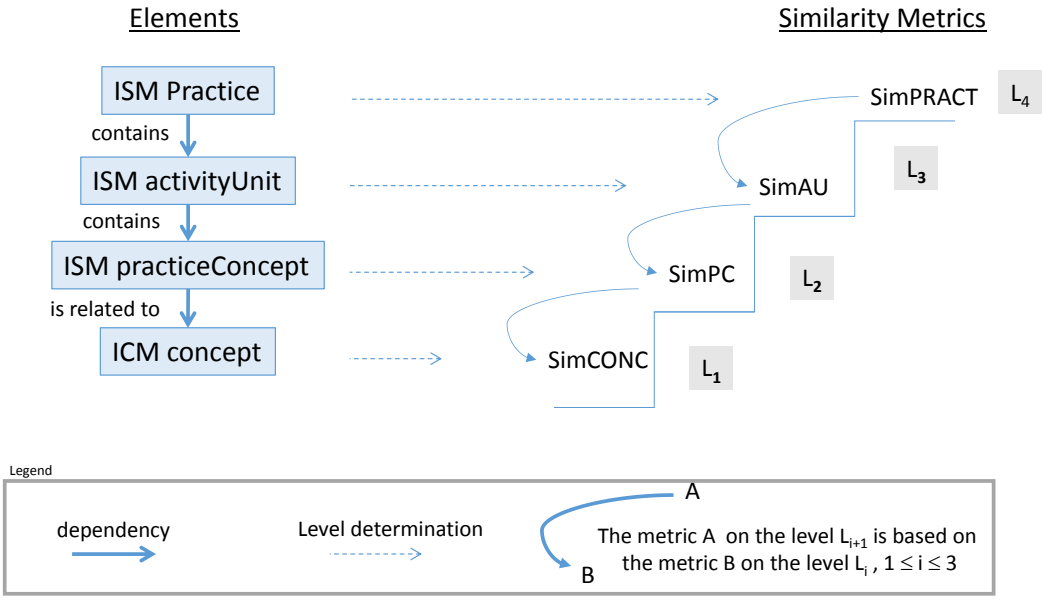


Fig. 30: Similarity metrics – Design

Firstly, each similarity metric depends on the similarity metric on the previous level. The similarity metric on the ISM practice level depends on the similarity metric on the ISM activityUnit level, that depends on the similarity metric on the ISM practiceConcept level, that in its turn depends on the similarity metric on the ICM concept level.

Secondly, the similarity degree on a level is high if the similarity degree on the previous level is also high. For example, ISM practices have a “High” similarity degree on the ordinal scale, if their parts, namely the ISM activityUnits, have a high similarity degree on the ratio scale. Analogously, ISM activityUnits have a high similarity degree on the ratio scale, if their ISM practiceConcepts have a high similarity degree on the ratio scale. Finally, the ISM practiceConcepts have a high similarity degree on the ratio scale, if their related ICM concepts have a high similarity degree on the ratio scale.

Finally, the similarity degree of ISM practices represents the final result.

5.5.2.1.1 Similarity Metrics for ICM concepts

For two concepts $conc_{ICM1}$ and a $conc_{ICM2}$, the metric **SimCONC (Similarity of Concepts)** determines their similarity degree. It is defined according to a ratio scale and depends on the type of hierarchy (generalizationOf-mono-hierarchy or composedOf-poly-hierarchy) the two ICM concepts belong to.

In a **generalizationOf-mono-hierarchy**, for each ICM concept there is only one ICM parent concept in this hierarchy.

In a **composedOf-poly-hierarchy**, an ICM part concept can have more ICM whole concepts in this hierarchy.

The metric is defined as follows:

$$SimCONC(conc_{ICM1}, conc_{ICM2}) = \begin{cases} 1, & \text{iff } conc_{ICM1} \text{ and } conc_{ICM2} \text{ refer to the same ICM concept.} \\ F_{genOf}(conc_{ICM1}, conc_{ICM2}), & \text{iff } conc_{ICM1} \text{ and } conc_{ICM2} \text{ are in the same generalizationOf-mono-hierachy.} \\ F_{comOf}(conc_{ICM1}, conc_{ICM2}), & \text{iff } conc_{ICM1} \text{ and } conc_{ICM2} \text{ are on one path in the same composedOf-poly-hierachy.} \\ F_{both}(conc_{ICM1}, conc_{ICM2}), & \text{iff } conc_{ICM1} \text{ is in a composedOf-poly-hierachy and } conc_{ICM2} \text{in a generalizationOf-mono-hierarchy.} \\ 0, & \text{otherwise.} \end{cases}$$

The functions F_{genOf}, F_{compOf} and F_{both} depend on different ICM structures. Two ICM concepts can be contained in a generalizationOf-mono-hierarchy, composedOf-poly-hierarchy or in both of them (Fig. 31).

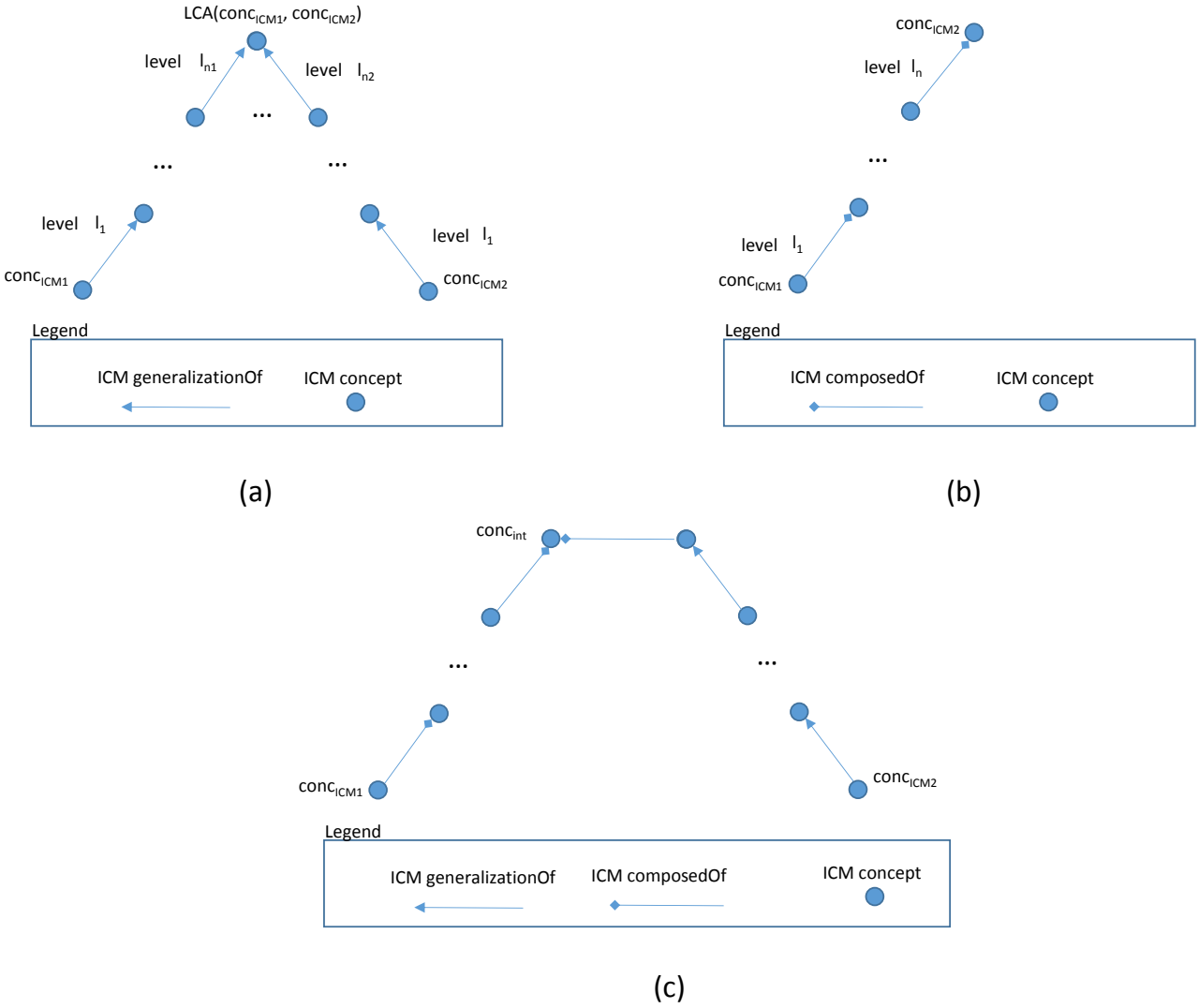


Fig. 31: ICM structures for the computation of the similarity between ICM concepts

F_{genOf} is defined for two ICM concepts contained in a generalizationOf-mono-hierarchy (Fig. 31 - (a)). It is based on the variant of the cosine distance method. The similarity of the $conc_{ICM1}$ and $conc_{ICM2}$ is high, if these are located deeply in the generalizationOf-mono-hierarchy and their lowest common ancestor (LCA) is located close to both of them. Furthermore, their similarity depends on the cohesion of the ICM conceptCategory ($cohesion_{cat}$) of an ICM concept on the level l_i and on the strength of the ICM generalizationOf ($strength_{genOf}$) between an ICM concept and its ICM parent concept on the level l_i in the generalizationOf-mono-hierarchy. There may be n_1 levels between $conc_{ICM1}$ and LCA and n_2 levels between $conc_{ICM2}$ and LCA. Therefore, the metric considers the cohesion and strength corresponding to the ICM concepts on the path between $conc_{ICM1}$ and LCA. Then, it considers the cohesion and strength corresponding to the ICM concepts on the path between $conc_{ICM2}$ and LCA and finally, the cohesion corresponding to the LCA itself.

$$F_{genOf}(conc_{ICM1}, conc_{ICM2}) = \prod_{i=1}^{n_1-1} \mathbf{Cohesion}_{\mathbf{cat}}(conc_{\mathbf{ICM1\_i}}) \cdot \mathbf{Strength}_{\mathbf{genOf}}(conc_{\mathbf{ICM1\_i}}, conc_{\mathbf{ICM\_i+1}}) \cdot \prod_{i=1}^{n_2-1} \mathbf{Cohesion}_{\mathbf{cat}}(conc_{\mathbf{ICM2\_i}}) \cdot \mathbf{Strength}_{\mathbf{genOf}}(conc_{\mathbf{ICM2\_i}}, conc_{\mathbf{ICM2\_i+1}}) \cdot \mathbf{Cohesion}_{\mathbf{cat}}(\mathbf{LCA}(conc_{\mathbf{ICM1}}, conc_{\mathbf{ICM2}})) \cdot \frac{2 \cdot \mathbf{depth}(\mathbf{LCA}(conc_{\mathbf{ICM1}}, conc_{\mathbf{ICM2}}))}{\mathbf{depth}(conc_{\mathbf{ICM1}}) + \mathbf{depth}(conc_{\mathbf{ICM2}})}$$

As the cohesion and the strength are defined on an ordinal scale, we define their corresponding rational values based on our experiences (Table 12). The rational values for the strength are defined based on an analysis of similar ISM practices and on the experts' evaluations of the similarity metrics results. The rational value for the cohesion is 0 or 1 as there are ICM concepts within an ICM conceptCategory that are to some point similar, but their similarity is so small that ISM practices that are connected to these ICM concepts should not be found as similar.

Strength - ICM GeneralizationOf	Value
High	1.00
Medium	0.85
Low	0.70

Cohesion - ICM conceptCategory	Value
High	1.00
Low	0.00

Table 12: Rational values for the ICM generalizationOf strength and ICM conceptCategory cohesion

F_{compOf} is defined for two ICM concepts contained in a composedOf-poly-hierarchy (Fig. 31 - (b)). It depends on the percentages ($Percentage_{compOf}$) of the ICM composedOf of an ICM concept on the level l_i. The $Percentage_{compOf}$ is given by the attribute percentage of the similarity relation ICM composedOf. Its value is set by the Modeler and is a ratio value between 0 and 1. There may be n levels l_i between $conc_{ICM1}$ and $conc_{ICM2}$ (l_1 is the level of $conc_{ICM1}$ and l_n is the level of $conc_{ICM2}$).

$$F_{compOf}(conc_{ICM1}, conc_{ICM2}) = \prod_{i=1}^{n} \mathbf{Percentage}_{\mathbf{compOf}}(conc_{\mathbf{ICM\_i}})$$

F_{both} is defined for two ICM concepts contained in both hierarchies (Fig. 31 - (c)). It multiplies the SimCONC values calculated according to the corresponding formulas until their intersections $conc_{int}$ (Fig. 31 – (c)).

$$F_{both}(conc_{ICM1}, conc_{ICM2}) = \mathbf{SimCONC}(conc_{ICM1}, conc_{ICM\_int}) \cdot \mathbf{SimCONC}(conc_{ICM2}, conc_{ICM\_int})$$

As the similarity metrics are to some extent difficult to understand, we give some examples for the computation of the SimCONC values for ICM concepts contained in the following ICM (Fig. 32).

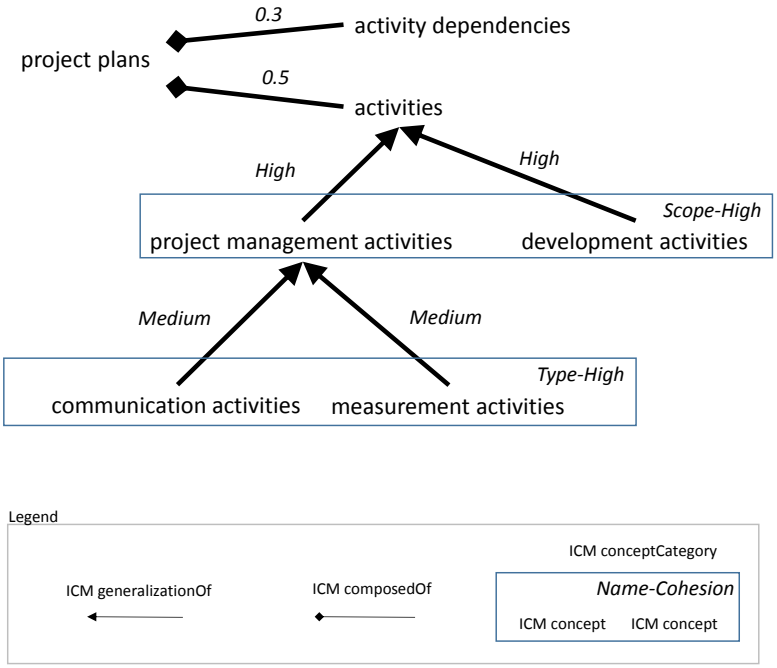


Fig. 32: Similarity metrics – Example ICM

To give interpretations for the similarity degree on this level, we map the ratio to an ordinal scale (Table 13).

Similarity degree	
Ratio scale	**Ordinal scale**
1.00	Equal
[0.67, 1.00)	High
[0.30, 0.67)	Medium
(0.00, 0.30	Low
0.00	Non-Equal

Table 13: Similarity degree – Mapping between a ratio scale and an ordinal scale

First, we calculate the similarity of the ICM concepts "project management activities" and "communication activities", as well as the similarity of "communication activities" and "measurement activities" that are in the same generalizationOf-mono-hierarchy. The similarity degree between these two pairs of ICM concepts is both "Medium". The first value is higher than the second value as the similarity degree between an ISM child concept and an ISM parent concept is higher than between ISM sibling concepts in the generalizationOf-mono-hierarchy.

$$\text{SimCONC}(\text{project management activities}, \text{communication activities})$$

$$= \text{High} \cdot \text{Medium} \cdot \frac{2\ \text{depth}\ (\text{project management activities})}{\text{depth}(\text{project management activities}) + \text{depth}(\text{communication activities})}$$

$$= 1 \cdot 0.85 \cdot \frac{2 * 2}{2 + 3} = 0.68$$

$$\text{SimCONC}(\text{measurement activities}, \text{communication activities})$$

$$= \text{High} \cdot \text{Medium} \cdot \text{Medium} \frac{2\ \text{depth (project management activities)}}{\text{depth}(\text{measurement activities}) + \text{depth}(\text{communication activities})}$$

$$= 1 \cdot 085 \cdot 0.85 \cdot \frac{2 * 2}{3 + 3} = 0.38$$

Furthermore, the similarity of the ICM concepts „project plans" and "activities" that are in the same composedOf-poly-hierarchy is given by the percentage the ICM part concept is part of the ICM whole concept:

$$\text{SimCONC}(\text{project plans}, \text{activities}) = 0.5$$

Finally, the similarity of the ICM concepts „project plans" and „ communication activities" which are related by ICM generalizationOf- and composedOf is 0.18. It means that the similarity degree between the considered ICM concepts is "Low".

$$\text{SimCONC}(\text{project plan}, \text{communication activities})$$

$$= \text{SimCONC}(\text{project plans}, \text{activities}) \cdot \text{SimCONC}(\text{activities}, \text{communication activities})$$

$$= 0.36 \cdot 0.5 = 0.18$$

5.5.2.1.2 Similarity Metrics for ISM PracticeConcepts

For the sets $PC_t = \{pc_{ISM\_t} \mid pc_{ISM\_t}$ is of type $t \in T\}$, the metric **$SimPC_t$ (Similarity of Practice Concepts)** determines the similarity degree of its ISM practiceConcepts of a certain type $t \in T$.

Let $CONC_{pcISM\_t} = \{conc_{ICM} \mid conc_{ICM}$ is related to $pc_{ISM\_t} \in PC_t\}$. This set contains all the ICM concepts that are related to the ISM practiceConcepts in PC_t.

Furthermore, there can be related ICM concepts that belong to a composedOf-poly-hierarchy and form a unit. We define the following sets to differentiate between these ICM concepts and the other related ICM concepts in generalizationOf-mono-hierarchies:

- $CONC_{whole} = \{conc_{ICM\_whole} \mid conc_{ICM\_whole} \in CONC_{pcISM\_t}$ and $conc_{ICM\_whole}$ is an ICM whole concept}
  This set contains all ICM whole concepts that are related to the ISM practiceConcepts in PC_t and form a unit.

- For a $conc_{ICM\_whole} \in CONC_{whole}$, its $UNITconc_{whole} = \{conc_{ICM\_part} \mid conc_{ICM\_part} \in CONC_{pcISM\_t}$ and $conc_{ICM\_part}$ is an ICM part concept of the ICM whole concept $conc_{ICM\_whole}\}$.
  This set contains all ICM part concepts for an ICM whole concept that are related to the ISM practiceConcepts in PC_t.

- $ALL\_UNIT = \bigcup_{conc_{ICM\_whole} \in CONCwhole} UNIT_{conc_{whole}} \cup CONC_{whole}$
  This set contains all ICM concepts that are related by ICM composedOf.

- NON_UNIT = $CONC_{pcISM\_t} \setminus ALL\_UNIT$
  This set contains the ICM concepts that are related by ICM generalizationOf.

Before introducing the metric $SimPC_t$, we exemplify the afore-mentioned sets for a better understanding. Based on the ICM structure presented in Fig. 32, let $CONC_{pcISM\_t}$ contain all the illustrated ICM concepts. For this set of ICM concepts, Table 14 illustrates the $CONC_{whole}$, $UNITconc_{whole}$ and NON_UNIT sets. As there is only one ICM whole concept, ALL_UNIT is equal to $UNITconc_{whole}$.

$CONC_{pcISM\_t}$	$CONC_{whole}$	$UNITconc_{whole}$	NON_UNIT
measurement activities, communication activities, project management activities, development activities, activities, activity dependencies, project plans	project plans	activities, activity dependencies	measurement activities, communication activities, project management activities, development activities, activities

Table 14: Similarity metrics – Example with ICM concepts that form a unit determined by ICM composedOf

$SimPC_t$ is defined according to a ratio scale and depends on the metric SimCONC for all ICM concepts related to the ISM practiceConcepts of PC_t. Furthermore, it uses two functions F_{unit} and $F_{non\text{-}unit}$ that considers the cases when there are only ICM concepts related by ICM composedOf or only ICM concepts related by ICM generalizationOf. The percentages of F_{unit} and $F_{non\text{-}unit}$ for $SimPC_t$ are calculated according to the number of ICM concepts that are relevant for each function. Finally, it considers the case when for a certain type (e.g. ISM role) there is only one ISM practiceConcept $\in$ PC_t. The metric is defined as follows:

$$SimPC_t(PC_t) = \begin{cases} F_{unit}(PC_t) \cdot \frac{|ALL\_UNIT|}{|CONC_{pcISM\_t}|} + F_{non\text{-}unit}(PC_t) \cdot \frac{|NON\_UNIT|}{|CONC_{pcISM\_t}|}, & \text{iff } |PC_t| > 1. \\ 0, & \text{iff } |PC_t| = 1. \end{cases}$$

F_{unit} aggregates the SimCONC values of the ICM concepts that form a unit. It is calculated as the average of the similarity of all such units. The similarity of an unit is calculated as the sum of all SimCONC values between the ICM part concepts and their ICM whole concept. Therefore, it has a high rational value if the SimCONC values are high.

$$F_{unit}(PC_t) = \frac{\sum_{conc_{whole} \in CONC_{whole}} [\sum_{conc_{ICM} \in UNIT_{conc_{whole}}} SimCONC(conc_{ICM}, conc_{whole})]}{|CONC_{whole}|}$$

The $F_{non\text{-}unit}$ is computed as the average of all SimCONC pairs ($conc_{ICMi}$, $conc_{ICMj}$), where for each ICM concept $conc_{ICMi}$, its highest ICM similar concept $conc_{ICMj}$ is considered. Therefore, it has a high rational value if the SimCONC values are high.

$$F_{non\text{-}unit}(PC_t) = \frac{\sum_{conc_{ICMi} \in NON\_UNIT} [MAX_{conc_{ICMj} \in NON\_UNIT} [SimCONC(conc_{ICMi}, conc_{ICMj})]]}{|NON\_UNIT|}$$

In the following, we exemplify the computation of F_{unit} and $F_{non\text{-}unit}$ for a set $PCt_{ISM\ output}$ that contains the following ISM outputs related to the ICM concepts illustrated in Fig. 32: "project plan", "activities", "activities dependencies", "communication activities", "project management activities" and "measurement activities".

To exemplify the computation of F_{unit}, we consider the ISM practiceConcepts "project plan", "activities" and "activities dependencies". Their ICM concepts with the same name form a single unit. Therefore, the SimPC value is computed as the sum of the SimCONC values between the ICM part concepts ("activities" and "activities dependencies") and their ICM whole concept ("project plan"). These SimCONC values are the percentages of the ICM composedOf between these ICM concepts.

$$F_{unit}(\text{project plans, activities, activity dependencies})$$

$$= \text{SimCONC}(\text{project plans, activities}) + \text{SimCONC}(\text{project plans, activity dependencies})$$

$$= 0.5 + 0.3 = 0.8$$

To exemplify the computation of $F_{non\text{-}unit}$, we consider the ISM practiceConcepts "communication activities", "project management activities" and "measurement activities". For each ISM practiceConcept, its best pair is searched. As already mentioned, the similarity degree between an ISM child concept and an ISM parent concept is higher than between ISM sibling concepts in the generalizationOf-mono-hierarchy. Therefore, for each ICM child concept its ICM parent concept is its best pair.

$$F_{non-unit}(\text{communication activities, project management activities, measurement activities})$$

$$= \frac{1}{2}[\text{SimCONC}(\text{communication activities, project management activities}) + \text{SimCONC}(\text{project management activities, measurement activities})]$$

$$= 0.68$$

The final result is computed as follows:

$$\text{SimPC}_{\text{ISM output}}(PC_{\text{ISM output}}) = \frac{3}{6} \cdot 0.8 + \frac{3}{6} \cdot 0.68 = 0.74$$

5.5.2.1.3 Similarity Metric for ISM ActivityUnits

For the set $AU = \{au_{ISMi} \mid i \geq 2\}$, the metric **SimAU (Similarity of ActivityUnits)** determines the similarity degree of all ISM activityUnits in AU.

We define the following sets:

- $PC_{auISM\_t} = \{pc_{ISM\_t} \mid pc_{ISM\_t}$ is contained in $au_{ISM} \in AU$ and pc_{ISM_t} is of type $t \in T\}$
  This set contains all ISM practiceConcepts of a certain type that are contained in the ISM activityUnits of AU.

- $T_{auISM} = \{t \mid t \in T$ and $\exists\ pc_{ISM\_t} \in PC_{auISM\_t}\}$
  This set contains all the types for which ISM practiceConcepts exist in the ISM activityUnits of AU.

SimAU is defined according to a ratio scale and depends on the metrics $SimPC_t$ for all ISM practiceConcepts of type $t \in T$ of the ISM activityUnits in AU. It is based on the weighted euclidian method and aggregates the $SimPC_t$ values according to the importance of the ISM practiceConcepts. The metric is defined as follows:

$$\boldsymbol{SimAU(AU)} = \sum_{t \in Type} \mathbf{Weight_t} \cdot \mathbf{SimPC_t}\ (\boldsymbol{PC_{auISM\_t}})$$

$$\boldsymbol{WEIGHT_t} = \frac{\boldsymbol{IMP_t}}{\sum_{type \in \mathbf{T_{auISM}}} \boldsymbol{IMP_{type}}}$$

5.5.2.1.4 Similarity Metrics for ISM Practices

For the set of practices PRACT = {$pract_{ISM\ i}$ | i ≥ 2}, the **SimPRACT (Similarity of Practices)** determines the similarity degree of all ISM practices in PRACT.

We define the following sets:

- $AU_{practISM}$ = {au_{ISM} | au_{ISM} is contained in $pract_{ISM}$ ∈PRACT}
  This set contains all ISM activityUnits that are contained in the ISM practices of PRACT.

- AU_{combi} = {{au_{ISM}} | {au_{ISM}} is a possible combination of size |PRACT| with $au_{ISM} \in AU_{practISM}$}
  This set contains all possible combinations of ISM activityUnits from each ISM practice in PRACT.

SimPRACT depends on the metric SimAU for all ISM activityUnits of the ISM practices in PRACT. It uses an ordinal scale as ISM practices are described differently in the various PRs. One ISM practice of a PR can contain only one ISM activityUnit, while an ISM practice of another PR can contain several ISM activityUnits. The aggregation of all SimAU values on a ratio scale could lead to low similarity values although the compared ISM practices contain ISM activityUnits with a high similarity degree. Therefore, the metric is defined as follows:

$\boldsymbol{SimPRACT(PRACT)} =$

„Equal“,	iff $\forall\{au_{ISM}\} \in AU_{combi}$, $SimAU(\{au_{ISM}\}) = 1$.
„High“,	iff $\exists\{au_{ISM}\} \in AU_{combi}$, $0.67 <= SimAU(\{au_{ISM}\}) < 1$.
„Medium“,	iff both conditions hold: • $\forall\{au_{ISM}\} \in AU_{combi}$, $SimAU(\{au_{ISM}\}) < 0.67$. • $\exists\{au_{ISM}\} \in AU_{combi}$, $SimAU(\{au_{ISM}\}) \geq 0.30$.
„Low“,	iff both conditions hold: • $\forall\{au_{ISM}\} \in AU_{combi}$, $SimAU(\{au_{ISM}\}) < 0.30$. • $\exists\{au_{ISM}\} \in AU_{combi}$, $SimAU(\{au_{ISM}\}) > 0$.
„Non-Equal“,	otherwise.

Two or more ISM practices with the similarity degree “Equal“, „High“, „Medium“, „Low“ or “Non-Equal” are called “Equal“, „High“, „Medium“, „Low“ and “Non-Equal” ISM practices respectively.

5.5.2.1.5 Requirements Verification

We verify the achievement of the requirements that the similarity metrics need to fulfill.

First, the general requirement for all metrics (R-All) requests that the metrics' results should be differentiable, comparable, reproducible and plausible. As the similarity metrics are based on the cohesion of ICM conceptCategories, on the ICM concepts and their similarity relations, the results are differentiable, comparable and reproducible. The plausibility of the similarity metrics are verified during several experiments performed with several experts (section 8.3).

The specific requirement (R1) requires that the importance of different ISM practiceConcepts types is considered. The similarity for ISM practices is based on the importance of its ISM practiceConcepts (attribute IMPt).

The specific requirement (R2) requires that the number of ISM practiceConcepts of an ISM activityUnit should not influence its similarity value. The similarity metrics only depend on the weight of the existing ISM practiceConcepts and their similarity. The weight for each ISM practiceConcept type is dynamically calculated based on the IMP_t attribute and does not depend on the number of ISM practiceConcepts.

5.5.2.2 Algorithm

The algorithm identifies the similarity between ISM practices. It considers as input the set of ISM practices PRACT = $\{pract_{ISMi} \mid i \geq 2\}$ and delivers as output the SimPRACT value.

The algorithm consists of the following steps:

1. Identify $AU_{practISM}$ = {au_{ISM} | au_{ISM} is contained in $pract_{ISM} \in$ PRACT}.
2. Identify AU_{combi} = {{au_{ISM}} | {au_{ISM}} is a possible combination of size |PRACT| with $au_{ISM} \in AU_{practISM}$}.
3. For each set {au_{ISM}} $\in AU_{combi}$ do:
   - **3.1.** Identify the sets PC_{auISM_t} = {pc_{ISM_t} | pc_{ISM_t} is contained in $au_{ISM} \in$ {au_{ISM}} and pc_{ISM_t} is of type $t \in T$}.
   - **3.2.** For each type $t \in T$ do:
     - **3.2.1.** Identify $CONC_t$ = {$conc_{ICM}$ | $conc_{ICM}$ is related to all $pc_{ISM\_t} \in PC_{auISM\_t}$}.

3.2.2. Identify all pairs (concICM1, concICM2) with concICM1 ∈ CONCt, concICM2 ∈ CONCt that correspond to ISM practiceConcepts from different ISM practices.

3.2.3. For each such pair (concICM1, concICM2) compute SimCONC(concICM1, concICM2).

3.2.4. Compute the SimPCt(PCauISM_t) based on the computed SimCONC values.

3.3. Compute the SimAU ({auISM}) based on the computed SimPCt values.

4. Compute SimPRACT(PRACT) based on the computed SimAU values.

5. Return SimPRACT(PRACT).

5.5.3 Identification of the Coverage between ISM practices

According to Goodman [Goodman 1972], "*Objects are similar if they have a set of common features*". Therefore, ISM practices are similar if they have a set of common elements that cover each other. We identify the coverage of two sets of ISM practices. Based on this coverage of two sets of ISM practices, other coverage results are possible. We give guidelines how to compute the highest coverage degree or the best coverage degree in a set of ISM practices.


Reminder

The **highest coverage** refers to a practice that has the maximum coverage degree in a considered set of practices.

The **best coverage** refers to the minimum subset of practices with a coverage degree of 1 in a considered set of practices.

For the identification of the coverage of ISM practices, we consider ISM practiceConcepts pc_{ISM_t} of type $t \in T$, $T = \{\text{ISM input, ISM output, ISM role}\}$. An ISM purpose is not relevant as the coverage is a measure for the adoption of elements. An ISM practice requests by its definition, only that its ISM outputs, ISM inputs and ISM roles are adopted.

5.5.3.1 Coverage Metrics

We define coverage metrics to determine the coverage degree of ISM practices.

According to the measurement theory, we have to define specific assumptions and requirements for these coverage metrics. As these are exactly the assumptions and requirements for the similarity metrics (see last section), we do not mention them anymore.

We also define the measurement scale for these coverage metrics. The coverage degree is given by a ratio scale with values between 0 and 1, where 1 is the highest value and means that the elements of the first set entirely covers the elements of the second set.

Furthermore, the coverage metrics are not symmetric: CovPRACT ({$pract_{ISM1}$}, {$pract_{ISM2}$, $pract_{ISM3}$}) ≠ CovPRACT ({$pract_{ISM2}$, $pract_{ISM3}$}, {$pract_{ISM1}$}).

According to our design principle, we define the coverage metrics on different levels (Fig. 33).

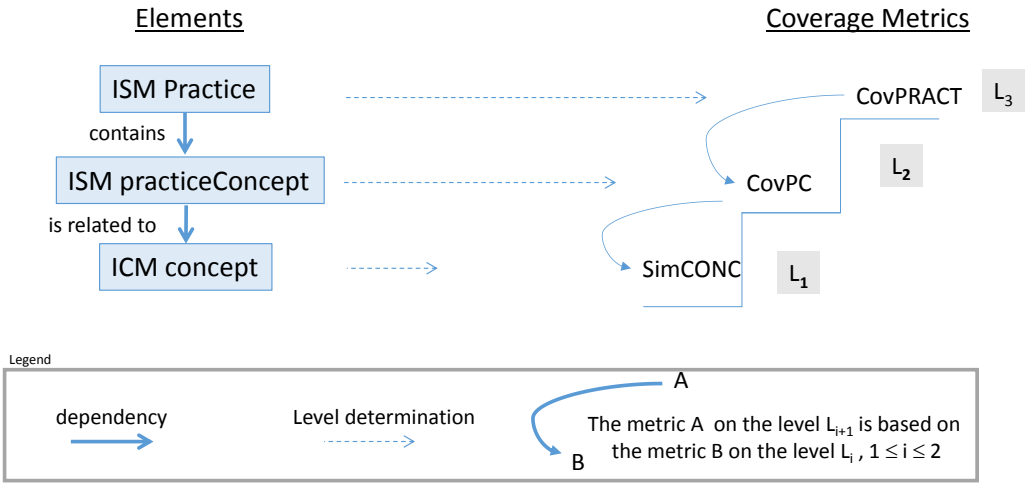


Fig. 33: Coverage metrics – Design

Firstly, each coverage metric depends on the coverage metric on the previous level. On the last level, it depends on the similarity metrics. The coverage metric on the ISM practice level depends on the coverage metric on the ISM practiceConcept level, that in its turn depends on the similarity metric on the ICM concept level.

Secondly, the coverage degree on a level is high on the given scale if the coverage or similarity degree on the previous level is high on the given scale. For example, the practice coverage of two sets of ISM practices is high, if the coverage degree of their ISM practiceConcepts is high. The coverage degree of the ISM practiceConcepts is high, if the ISM practiceConcepts of the first set are highly covering the ISM practiceConcepts of the second set. The ISM practiceConcepts highly cover other ISM practiceConcepts, if these are highly similar to each other, i.e. if the similarity degree between their ICM concepts is high.

Finally, the coverage degree between ISM practices represents the final result.

5.5.3.1.1 Coverage Metrics for ISM PracticeConcepts

For the type $t \in T$ and the sets $PC_{t\_1} = \{pc_{ISM\_t} \mid pc_{ISM\_t}$ is of type t} and $PC_{t\_2} = \{pc_{ISM\_t} \mid pc_{ISM\_t}$ is of type t}, the metric **$CovPC_t$ (Coverage of Practice Concepts)** defines the coverage degree between PC_{t_1} and PC_{t_2}.

We define the following sets:

- $CONC_{pcISM\_t\_1} = \{conc_{ICM} \mid conc_{ICM}$ is related to $pc_{ISM\_t} \in PC_{t\_1}\}$
  This set contains all the ICM concepts that are related to the ISM practiceConcepts in PC_{t_1}.

- $CONC_{pcISM\_t\_2} = \{conc_{ICM} \mid conc_{ICM}$ is related to $pc_{ISM\_t} \in PC_{t\_1}\}$
  This set contains all the ICM concepts that are related to the ISM practiceConcepts in PC_{t_2}.

Analogously to the similarity metrics, there are related ICM concepts that belong to a composedOf-poly-hierarchy and form a unit. We remind about the ICM whole and part concepts:

An **ICM whole concept** and **ICM part concept** are related by an ICM composedOf. An ICM whole concept can be composedOf more ICM part concepts. For example, the ICM whole concept "project plans" is composedOf ICM part concepts "activities" and "activity dependencies".

ICM whole concepts entirely cover its ICM part concepts. If $CONC_{pcISM\_t\_1}$ contains such ICM whole concepts and $CONC_{pcISM\_t\_2}$ contains their ICM part concepts, then $CONC_{pcISM\_t\_1}$ covers a part of $CONC_{pcISM\_t\_2}$. We define the following sets to differentiate between ICM whole and part concepts in the composedOf-poly-hierarchies and ICM concepts in generalizationOf-mono-hierarchies:

- $CONC_{whole} = \{conc_{ICM\_whole} \mid conc_{ICM\_whole} \in CONC_{pcISM\_t\_1}$ and $conc_{ICM}$ is an ICM whole concept of an ICM part concept in $CONC_{pcISM\_t\_2}\}$
  This set contains all ICM concepts of the ISM practiceConcepts in PC_{t_1} that are ICM whole concepts of ICM part concepts of the ISM practiceConcepts in PC_{t_2}.

- For a $conc_{ICM\_whole} \in CONC_{whole}$, its $UNITconc_{whole} = \{conc_{ICM\_part} \mid conc_{ICM\_part} \in CONC_{pcISM\_t\_2}$ and $conc_{ICM\_part}$ is an ICM part concept of the ICM whole concept $conc_{ICM\_whole}\}$.
  This set contains all ICM part concepts of the ISM practiceConcepts in PC_{t_2} for an ICM whole concept.

- $ALL\_UNIT\_2 := \bigcup_{conc_{ICM\_whole} \in CONCwhole} UNIT_{conc_{whole}}$
  This set contains all ICM concepts of the ISM practiceConcepts in PC_{t_2} that are related by ICM composedOf.

- $NON\_UNIT\_2 = CONC_{pcISM\_t\_2} \setminus ALL\_UNIT\_2$
  This set contains the ICM concepts of the ISM practiceConcepts in PC_{t_2} that are not ICM part concepts.

- $NON\_UNIT\_1 = CONC_{pcISM\_t\_1} \setminus CONC_{whole}$
  This set contains all ICM concepts of the ISM practiceConcepts in PC_{t_1} that are that are not ICM whole concepts.

Before introducing the metric $CovPC_t$, we exemplify the afore-mentioned sets for a better understanding. Let $CONC_{pcISM\_t\_1}$ and $CONC_{pcISM\_t\_2}$ be defined as in Fig. 34.

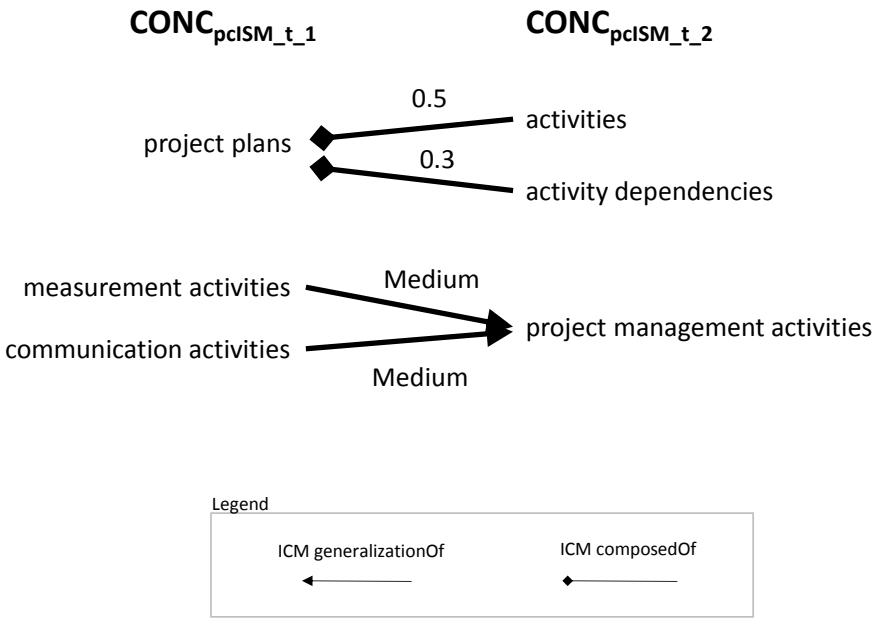


Fig. 34: Coverage metrics – Examples of two sets of ICM concepts

Table 15 illustrates the $CONC_{whole}$, $UNITconc_{whole}$ and NON-UNIT sets. As there is only one ICM whole concept, ALL-UNIT_2 is equal to $UNITconc_{whole}$.

$CONC_{pcISM\_t\_1}$	$CONC_{pcISM\_t\_2}$	$CONC_{whole}$	$UNITconc_{whole}$	NON_UNIT_1	NON_UNIT_2
project plans, measurement activities, communication activities	activities, activity dependencies, project management activities	project plans	activities, activity dependencies	measurement activities, communication activities	project management activities

Table 15: Coverage metrics – Example with ICM concepts that form a unit determined by ICM composedOf

$CovPC_t$ depends on the metric SimCONC for each type $t \in T$. Analogously to the similarity metrics on the ISM practiceConcept level, it uses F_{unit} and $F_{non\text{-}unit}$ that consider the cases when there are only ICM concepts related by ICM composedOf and generalizationOf respectively. The function F_{unit} is actually a constant and always has the value 1. This is because ICM whole concepts from $CONC_{pcISM\_t\_1}$ entirely cover the ICM part concepts from $CONC_{pcISM\_t\_2}$. The percentages of the two functions for $CovPC_t$ are calculated according to the number of ICM concepts in $CONC_{pcISM\_t\_2}$ that are relevant for each function. The number of ICM concepts in $CONC_{pcISM\_t\_1}$ is not relevant as these ICM concepts do not influence the coverage degree. Finally, it considers the case when for a certain type (e.g. ISM role) there are no ISM practiceConcept $\in PC_{t\_1}$. It is defined as follows:

$$CovPC_t(PC_{t\_1}, PC_{t\_2}) = \begin{cases} \frac{|ALL\_UNIT\_2|}{|CONC_{pcISM\_t\_2}|} + F_{non\text{-}unit}(PC_{t\_1}, PC_{t\_2}) \cdot \frac{|NON\_UNIT\_2|}{|CONC_{pcISM\_t\_2}|}, & \text{iff } |PC_{t\_1}| > 0. \\ 0, & \text{iff } |PC_{t\_1}| = 0. \end{cases}$$

The $F_{non\text{-}unit}$ is computed as the average of all SimCONC pairs ($conc_{ICMi}$, $conc_{ICMj}$), where for each ICM concept $conc_{ICMi}$ from $CONC_{pcISM\_t\_2}$, its highest ICM similar concept $conc_{ICMj}$ from $CONC_{pcISM\_t\_1}$ is considered. This means that, for the $conc_{ICMi}$ we search for $conc_{ICMj}$ that best covers it to compute the coverage of $CONC_{pcISM\_t\_1}$ in $CONC_{pcISM\_t\_2}$. Consequently, this coverage has a high rational value if the SimCONC values are high.

$$F_{non\text{-}unit}(PC_{t\_1}, PC_{t\_2}) = \frac{\sum_{conc_{ICMi} \in \text{NON\_UNIT\_2}} [\mathbf{MAX}_{conc_{ICMj} \in \text{NON\_UNIT\_1}} [\mathbf{SimCONC}(conc_{ICMi}, conc_{ICMj})]]}{|\text{NON\_UNIT\_2}|}$$

As the coverage metrics on this level are to some extent difficult to understand, we exemplify the computation of $CovPC_{ISM\ output}$ for ISM practiceConcepts of type ISM output. We consider the two sets of ISM practiceConcepts (Fig. 34). The result is 0.89 and it means that the first set of ISM practiceConcepts highly covers the second set of ISM practiceConcepts.

$$\begin{aligned} &\text{CovPC}_{ISM\ output}(\text{PC}_{\text{ISM output\_1}}, \text{PC}_{\text{ISM output\_2}}) \\ &= \frac{2}{3} + \text{SimCONC(project management activities, measurement activities)} \cdot \frac{1}{3} \\ &= \frac{2}{3} + 0.68 \cdot \frac{1}{3} = 0.89 \end{aligned}$$

5.5.3.1.2 Coverage Metrics for ISM Practices

For two sets $PRACT_1 = \{pract_{ISMi} \mid i \geq 1\}$ and $PRACT_2 = \{pract_{ISMi} \mid i \geq 1\}$, the metric **CovPRACT (Coverage of Practices)** calculates the coverage degree between $PRACT_1$ and $PRACT_2$.

We define the following sets:

- $T_{practISM} = \{t \mid t \in T \text{ and } \exists\ pc_{ISM\_t} \in PC_{practISMt\_2}\}$
  This set contains all the types for which ISM practiceConcepts exist in the ISM practices of $PRACT_2$.

- $PC_{practISM\_t\_1} = \{pc_{ISM\_t} \mid pc_{ISM\_t}$ is contained in $pract_{ISM} \in PRACT_1$ and pc_{ISM_t} is of type $t \in T_{practISM}\}$
  $PC_{practISM\_t\_2} = \{pc_{ISM\_t} \mid pc_{ISM\_t}$ is contained in $pract_{ISM} \in PRACT_2$ and pc_{ISM_t} is of type $t \in T_{practISM}\}$
  These sets contain all ISM practiceConcepts of the same type that are contained in the ISM practices of $PRACT_1$ or of $PRACT_2$ respectively.

CovPRACT depends on the metric $CovPC_t$ for each type $t \in T_{practISM}$. It is based on the variant of the weighted euclidian distance and aggregates the $CovPC_t$ values according to the importance of the ISM practiceConcepts. The metric is defined as follows:

$$CovPRACT(PRACT_1, PRACT_2) = \sum_{t \in Type} [\mathbf{Weight_t} \cdot \mathbf{CovPC_t}(PC_{practISM\_t\_1}, PC_{practISM\_t\_2})]$$

$$WEIGHT_t = \frac{IMP_t}{\sum_{type \in \mathbf{T_{practISM}}} IMP_{type}}$$

5.5.3.1.3 Requirements Verification

We verify the achievement of the requirements that the coverage metrics need to fulfill.

First, the general requirement for all metrics (R-All) requests that the metrics' results should be differentiable, comparable, reproducible and plausible. The first three conditions are true as the coverage metrics are based on the similarity metrics that fulfill this requirement. The plausibility of the coverage metrics are verified only by us as we could not involve experts in this evaluation. However, the coverage metrics are based on the similarity metrics and thus, their plausibility is partially verified.

The specific requirement (R1) requires that the importance of the different ISM practiceConcepts types is considered. The coverage metrics for ISM practices are based on the importance of its ISM practiceConcepts (attribute IMPt).

The specific requirement (R2) requires that the number of ISM practiceConcepts of an ISM activityUnit should not influence the coverage value. The coverage metrics only depends on the weight of the existing ISM practiceConcepts and their coverage. The weight for each ISM practiceConcept type is dynamically calculated based on the IMP_t attribute and does not depend on the number of ISM practiceConcepts.

5.5.3.2 Algorithm

The algorithm identifies the coverage of two sets of ISM practices, i.e. the coverage degree of the first set of ISM practices in the second set of practices. It considers as input the two sets of ISM practices: $PRACT_1 = \{pract_{ISMi} \mid i \geq 1\}$ and $PRACT_2 = \{pract_{ISMi} \mid i \geq 1\}$. The algorithm delivers as output their CovPRACT value.

The algorithm consists of the following steps:

1. For each type t ∈ T do:
   - **1.1.** Identify the sets PCt_1 = {pcISM_t | pcISM_t is contained in all practISM ∈ PRACT1 and pcISM_t is of type t ∈ T}.
   - **1.2.** Identify the sets PCt_2 = {pcISM_t | pcISM_t is contained in all practISM ∈ PRACT2 and pcISM_t is of type t ∈ T}.
   - **1.3.** Compute the CovPCt(PCt_1, PCt_2) based on the computed SimCONC values.
2. Compute CovPRACT (PRACT1, PRACT2).
3. Return CovPRACT (PRACT1, PRACT2).

Based on the coverage metrics, we can also calculate the *highest coverage degree* and the *best coverage degree of a set of ISM practices*.

To calculate the *highest coverage degree of a set of ISM practices*, we compute the coverage degree for each ISM practice within this set of ISM practices and determine the maximum CovPRACT value.

To calculate the *best coverage degree of a set of ISM practices*, we compute the coverage degree for each combination of ISM practices with one, two or more ISM practices in the set of ISM practices. Then, we identify if there is a combination with a minimum number of ISM practices that has a coverage degree of 1.

5.5.4 Identification of the Output States of ISM Practices

According to Goodman [Goodman 1972], "*Objects are similar if they have a set of common features*". Therefore, ISM practices are similar if they are related to common ISM outputs that are similar.

For the identification of the output states between ISM practices, we consider ISM practiceConcepts pc_{ISM_t} of type $t \in T$, $T = \{\text{ISM output}\}$. As the name says, we compute the output states of the ISM practices and thus, consider only ISM outputs.

5.5.4.1 Output State Metrics

We define the output state metrics to determine the output state of an ISM practice regarding its ISM outputs.

According to the measurement theory, we define the following specific assumption (A1). Besides the general requirement (R-All), we do not specify any other requirement for these metrics:

A1. Following states of an ICM concept have a time order: "Plan"< "Do"< "Check" < "Do-Check", where the "Plan" is the earliest state and "Do-Check" is the latest state on the timeline.

The assumption (A1) defines the time order of the states. As the state "Plan" reflects the creation of an element, the state "Do" reflects its implementation, "Check" its verification after the implementation and Do-Check" its verification and update after this verification.

According to the measurement theory, we define the measurement scale. Based on the assumption A1, the output state metrics use an ordinal scale with the order: "Plan" < "Do" < "Check" < "Do-Check", where the "Plan" is the earliest state and "Do-Check" the latest state on the timeline.

According to our design principle, we define the output state metrics on different levels (Fig. 35).

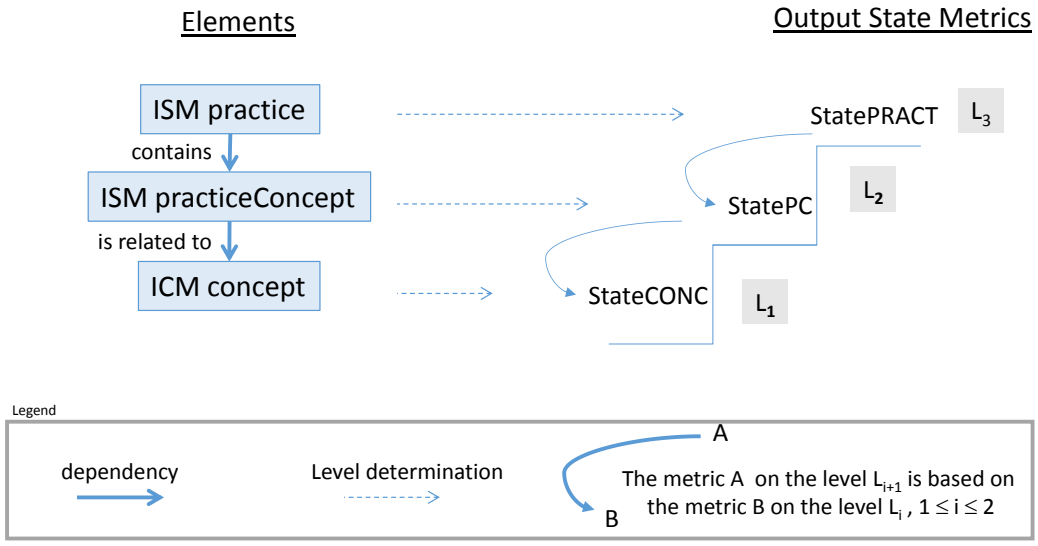


Fig. 35: Output state metrics – Design

Firstly, each output state metric depends the output state metric on the previous level. The output state metric on the ISM practice level depends on the output state metric on the ISM practiceConcept level, that depends on the output state metric on the ICM concept level.

Secondly, the output state on a level is high on the given scale if the output state on the previous level is also high on the given scale. For example, the output state of an ISM practice is "Do-Check", if the output state of one of its ISM practiceConcepts is "Do-Check". Analogously, the output state of the ISM practiceConcepts is "Do-Check", if the output state of the corresponding ICM concepts is "Do-Check".

Finally, the output state of the ISM practices represents the final result.

5.5.4.1.1 Output State Metrics for ICM Concepts

For a $conc_{ICM}$, the metric **StateCONC** (**State of Concepts**) determines its output state.

StateCONC depends on the different states of the ICM concept $conc_{ICM}$ that can be extracted from its ICM conceptCategory. It is defined as follows:

$$StateCONC(conc_{ICM}) = \begin{cases} \textbf{\textit{„Do“}}, & \text{iff the ICM conceptCategory of } conc_{ICM} \text{ contains „Do“.} \\ \textbf{\textit{„Check“}}, & \text{iff the ICM conceptCategory of } conc_{ICM} \text{ contains „Check“ but it does not contain „Do“.} \\ \textbf{\textit{„Do-Check“}}, & \text{iff both conditions hold:} \\ & \bullet \text{ ICM conceptCategory of } conc_{ICM} \text{ contains „Do“.} \\ & \bullet \text{ ICM conceptCategory of } conc_{ICM} \text{ contains „Check“.} \\ \textbf{\textit{„Plan“}}, & \text{otherwise.} \end{cases}$$

5.5.4.1.2 Output State Metrics for ISM practiceConcepts

For $pc_{ISM\_output,}$ the metric **StatePC** (**State of Practice Concepts**) determines its output state.

Let $CONC_{pcISM\_output} = \{conc_{ICM} \mid conc_{ICM}$ is related to $pc_{ISM\_output}\}$. This set contains all ICM concepts related to pc_{ISM_output}.

StatePC depends on the metric StateCONC for the ICM concepts related to pc_{ISM_output}. The metric is computed as the latest value of the ordinal scale: "Plan" < "Do" < "Check" < "Do-Check". It has the value "Do" if there exists a StateCONC value that is "Do", but no "Check" or "Do-Check". It has the value "Check" if there exists a "Check", but no "Do-Check". It has the value "Do-Check" if there exists "Do-Check". Otherwise the value is "Plan". The metric is defined as follows:

$$StatePC(pc_{ISM\_output}) =$$

- ***„Do“***, iff the two conditions hold:
  - $\exists\ conc_{ICM} \in CONC_{pcISM}$, $StateCONC(conc_{ICM}) =$ „Do“.
  - $\nexists\ conc_{ICM} \in CONC_{pcISM}$, $SupCONC(conc_{ICM}) \in \{$„Check“ , Do-Check“$\}$.
- ***„Check“***, iff the two conditions hold:
  - $\exists\ conc_{ICM} \in CONC_{pcISM}$, $StateCONC(conc_{ICM}) =$ „Check“.
  - $\nexists\ conc_{ICM} \in CONC_{pcISM}$, $SupCONC(conc_{ICM}) =$ „Do-Check“.
- ***„Do-Check“***, iff $\exists\ conc_{ICM} \in CONC_{pcISM}$, $StateCONC(conc_{ICM}) =$ „Do-Check“.
- ***„Plan“***, otherwise.

5.5.4.1.3 Output State Metrics for ISM Practices

For $pract_{ISM,}$ the metric **StatePRACT** (**State of Practices**) determines its output state.

Let $PC_{practISM} = \{pc_{ISM\_output} \mid pc_{ISM\_output}$ is contained in $pract_{ISM}\}$. This set contains all ISM practiceConcepts contained in $pract_{ISM.}$

StatePRACT is depends on the metric StatePC for the ISM practiceConcepts contained in $pract_{ISM}$. It is computed analogously to the metric StatePC as the latest value of the ordinal scale: "Plan" < "Do" < "Check" < "Do-Check":

$$StatePRACT(pract_{ISM}) =$$

- ***„Do“***, iff the two conditions hold:
  - $\exists\ pc_{ISM\_output} \in PC_{practISM}$, $StatePC(pc_{ISM\_output}) =$ „Do“.
  - $\nexists\ pc_{ISM\_output} \in PC_{practISM}$, $StatePC(pc_{ISM\_output}) \in \{$„Check“ , Do-Check“$\}$.
- ***„Check“***, iff the two conditions hold:
  - $\exists\ pc_{ISM\_output} \in PC_{practISM}$, $StatePC(pc_{ISM\_output}) =$ „Check“.
  - $\nexists\ pc_{ISM\_output} \in PC_{practISM}$, $StatePC(pc_{ISM\_output}) =$ „Do-Check“.
- ***„Do-Check“***, iff $\exists\ pc_{ISM\_output} \in PC_{practISM}$, $StatePC(pc_{ISM\_output}) =$ „Do-Check“.
- ***„Plan“***, otherwise.

One ISM practice with the output state “Plan“, „Do“, „Check“ or „Do-Check“ is called Plan“, „Do“, „Check“ or „Do-Check“ ISM practice respectively.

5.5.4.1.4 Requirements Verification

We verify the achievement of the general requirement (R-All) that the output state metrics need to fulfill. As these metrics are based on the ICM conceptCategory of an ICM concept, the results are differentiable, comparable and reproducible. The plausibility of the output states metrics are verified only by us as we could not involve any experts in this evaluation.

5.5.4.2 Algorithm

The algorithm identifies the output states of ISM practices. It considers as input an ICM concept $conc_{ICM}$ and delivers as output the ISM practices and their StatePRACT values.

The algorithm consists of the following steps:

1. Let STATE_PRACT ={}.

2. For concICM identify the CONCD = {concICM | concICM is an ICM descendant concept of concICM}.
Let CONC = {concICM} ∪ CONCD.

3. For each concICM ∈ CONC identify PCconcICM = {pcISM_t | pcISM_t is related to concICM and pcISM_t is of type t ∈ T}.

4. For each pcISM_t ∈ PCconcICM do:

4.1. Identify PRACTpcISM = {practISM | practISM contains pcISM }.

4.2. For each practISM ∈ PRACTpcISM do:

4.2.1. Identify PCpractISM_t = {pcISM_t | pcISM_t is contained in practISM and pcISM_t is of type t ∈ T}.

4.2.2. For each pcISM_t ∈PCpractISM_t do:

4.2.2.1. Identify CONCpcISM_t = {concICM | concICM is related to pcISM_t}.

4.2.2.2. For each concICM ∈ CONCpcISM_t compute the StateCONC(concICM).

4.2.2.3. Compute the StatePC(pcISM_t) based on the computed StateCONC values.

4.2.3. Compute the StatePRACT (practISM) based on the computed StatePC values.

4.2.4. Let STATE_PRACT = STATE_PRACT ∪ {(practISM, StatePRACT(practISM)}.

5. Return STATE_PRACT.

5.5.5 Examples

We illustrate the identification of the following similar ISM practices:

- CMMI-DEV PPQA SP2.1 "Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers."
- SPICE SUP.1 BP9 "Ensure resolution on non-conformances."

The identification of similar ISM practices is based on the relations between the ISM practiceConcepts and ICM concepts. Fig. 29 illustrates the modeling of the CMMI-DEV PPQA SP2.1 and SPICE SUP.1 BP9 practices and their related elements.

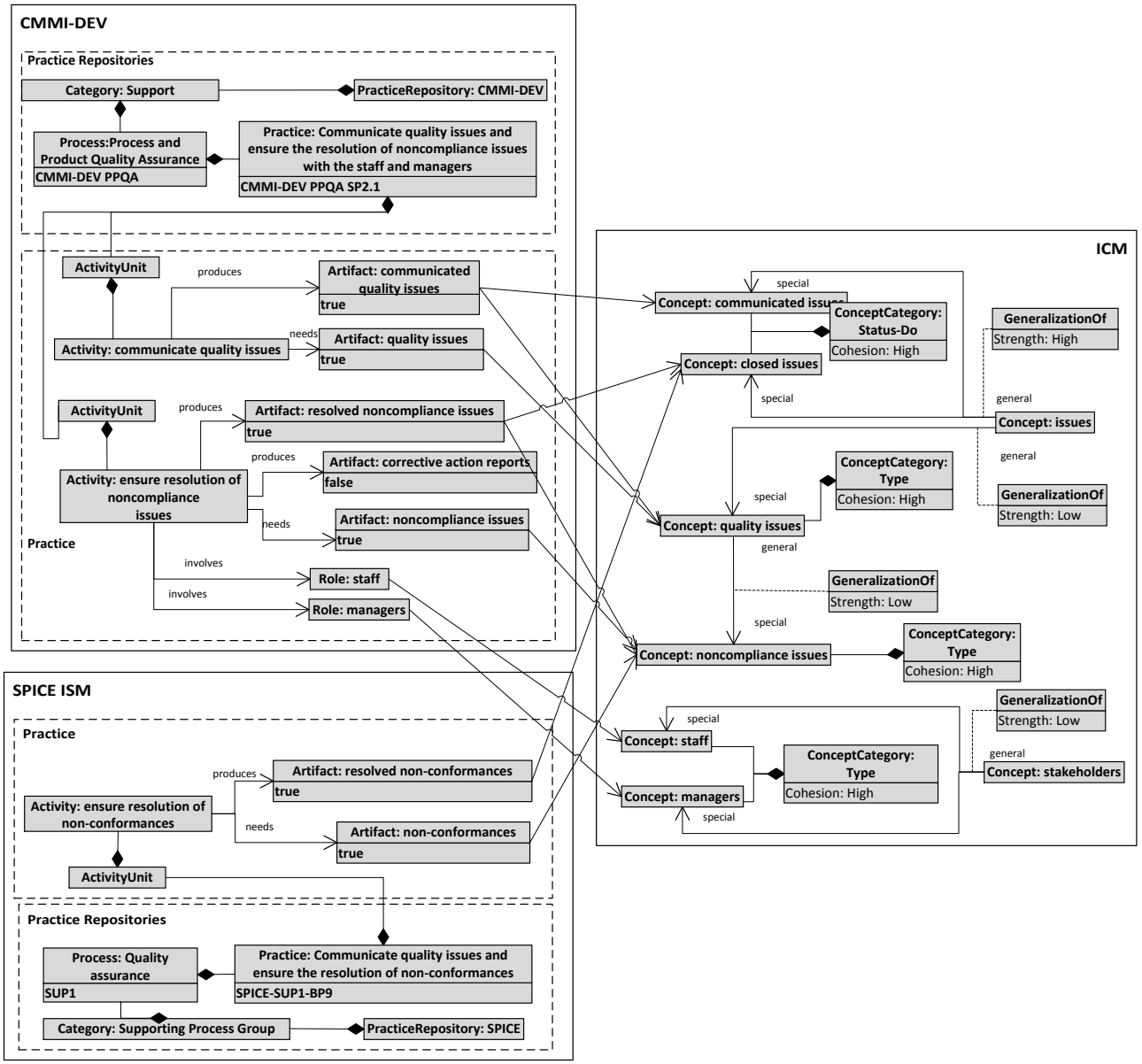


Fig. 36: Example – Identification of similar ISM practices – Excerpts of ISMs and ICM

Table 16 illustrates the steps to compute the similarity degrees on the different levels. As the similarity degree computation does not consider ISM implicit artifacts, the ISM practice element "corrective action reports" is not considered.

First, all combinations of ISM activityUnits are identified. For each set of ISM activityUnits, the SimAU is computed based on the SimPC values of the ISM practiceConcepts in such a combination of ISM activityUnits. For each type, the $SimPC_t$ is computed based on the SimCONC values of the corresponding ICM concepts. The similarity degree between the ISM practices is "High" as there is one ISM activityUnit combination that has a high similarity degree, namely the value 0.83.

Table 17 illustrates the steps to compute the coverage degrees on the different levels. First, the coverage of the CMMI-DEV practice is computed. Secondly, the coverage of the SPICE practice is computed. For this purpose, the CovPRACT values are computed based on the CovPC values of their

ISM practiceConcepts. The ISM practiceConcepts of the CMMI-DEV practice covers the ISM practiceConcepts of the SPICE practice. The ISM practiceConcepts of the SPICE practice do not entirely cover the ISM practiceConcepts of the CMMI-DEV practice. Consequently, the CMMI-DEV practice has the highest coverage degree.

Finally, Table 18 illustrates the steps performed to compute the output states on the different levels.

Table 16: Example – Similarity Algorithm

Input	Step 1	Step 2	Step 3							Step 4 (Output)
			Step 3.1	Step 3.2				Step 3.3		
				Step 3.2.1	Step 3.2.2	Step 3.2.3		Step 3.2.4		
ISM practices (PRACT)	ISM ActivityUnits ($AU_{practISM}$)	ISM ActivityUnits (AUcombi)	ISM practiceConcepts (PC_{aulSM_t})	ICM concept ($CONC_t$)	ICM concepts pairs	SimCONC		SimPC	IMP	SimAU	SimPract
{CMMI-DEV PPQA SP2.1, SPICE SUP.1.BP9}	$AU_{CMMI\text{-}DEV\ PPQA\ SP2.1}$ = {communicate quality issues, ensure resolution of noncompliance issues} $AU_{SPICE\ SUP.1.BP9}$ = {ensure resolution of non-conformances}	{communicate quality issues, ensure resolution of non-conformances}	{communicated quality issues, resolved non-conformances} Type: ISM output	{communicated issues, closed issues, noncompliance issues, quality issues}	(communicated issues, closed issues)	$\frac{1 \cdot 1 \cdot 2}{(2+2)}$	0.5	0.53	0.74	0.54	High
					(communicated issues, noncompliance issues)	$\frac{0.7 \cdot 0.7 \cdot 2}{(2+3)}$	0.28				
					(quality issues, closed isses)	$\frac{0.7 \cdot 2}{(2+2)}$	0.35				
					(quality issues, noncompliance issues)	$\frac{0.7 \cdot 2 \cdot 2}{(2+3)}$	0.56				
			{quality issues, non-conformances} Type: ISM input	{quality issues, noncompliance issues}	(quality issues, noncompliance issues)	$\frac{0.7 \cdot 2 \cdot 2}{(2+3)}$	0.56	0.56	0.26		
		{ensure resolution of noncompliance issues, ensure resolution of non-conformances}	{resolved noncompliance issues, resolved non-conformances} Type: ISM output	{noncompliance issues, closed issues}	(closed issues, closed issues)	1	1.0	1	0.61	0.83	
					(noncompliance issues, noncompliance issues)	1	1.0				
			{noncompliance issues, non-conformances} Type: ISM input	{noncompliance issues, noncompliance issues}	(noncompliance issues, noncompliance issues)	1	1.0	1	0.22		
			{staff, managers} Type: ISM role	{staff, managers}	(staff, o)	0	0.0	0	0.17		
					(managers, o)	0	0.0				

Table 17: Example – Coverage Algorithm

Input	Step 1								Step 2
	Step 1.1	Step 1.2					Step 1.3		
ISM Practices (PRACT1 and PRACT2)	ISM practiceConcepts (PCt_1)	ISM practiceConcepts (PCt_2)	Type	ICM concept (CONCpcISM_t_1)	ICM concept (CONCpcISM_t_2)	IMP	CovPC		CovPRACT
PRACT1={CMMI-DEV PPQA SP2.1} PRACT2 = {SPICE SUP1.BP9}	{communicated quality issues, resolved noncompliance issues}	{resolved non-conformances}	ISM output	{communicated issues, closed issues, noncompliance issues, quality issues}	{closed issues, noncompliance issues}	0.74	$\frac{1+1}{2}$	1	1
	{quality issues, noncompliance issues}	{non-conformances}	ISM input	{noncompliance issues, quality issues}	{noncompliance issues}	0.26	1	1	
	{staff, managers}	{o}	ISM role	{staff, managers}	{o}	0.00	0	0	
PRACT1={SPICE SUP1.BP9} PRACT2 = {CMMI-DEV PPQA SP2.1}	{resolved non-conformances}	{resolved non-conformances, communicated quality issues, resolved noncompliance issues}	ISM output	{closed issues, noncompliance issues}	{communicated issues, closed issues, noncompliance issues, quality issues}	0.61	$\frac{0.5+1+1+0.56}{4}$	0.77	0.64
	{non-conformances}	{non-conformances, quality issues, noncompliance issues}	ISM input	{noncompliance issues}	{noncompliance issues, quality issues}	0.22	$\frac{1+0.56}{2}$	0.78	
	{o}	{staff, managers}	ISM role	{o}	{staff, managers}	0.17	0	0	

Table 18: Example – Output States Algorithm

Input	Step 2	Step 3	Step 4	Step 4.2						
			Step 4.1	Step 4.2.1	Step 4.2.2			Step 4.2.3	Step 4.2.4	
					Step 4.2.2.1	Step 4.2.2.2	Step 4.2.2.3			
ICM concept	**ICM concept incl. ICM descendant concepts (CONC)**	**ISM practice elements ($PC_{concICM}$)**	**ISM Practices ($PRACT_{pcISM\_output}$)**	**ISM practiceConcepts in $PC_{practISM}$**	**ICM concepts in $CONC_{pcISM\_output}$**	**StateCONC**	**StatePC**	**StatePRACT**	**STATE_PRACT**	
quality issues	{quality issues, noncompliance issues}	{communicated quality issues, resolved noncompliance issues, resolved nonconformances}	{CMMI-DEV PPQA SP2.1, SPICE SUP.1.BP9}	communicated quality issues	communicated issues	Do	Do	Do	{(CMMI-DEV PPQA SP2.1, Do), (SPICE SUP.1.BP9, Do)}	
					quality issues	Plan				
				resolved noncompliance issues	resolved issues	Do	Do			
					noncompliance issues	Plan				
				quality issues	quality issues	Plan	Plan			
				noncompliance issues	noncompliance issues	Plan	Plan			
				non-conformances	noncompliance issues	Plan	Plan	Do		
				resolved non-conformances	resolved issues	Do	Do			
					noncompliance issues	Plan				

5.6 Identification of ISM Practice Dependencies

In this section we describe how MOSAIC implements the identification of ISM practice dependencies. We propose the dependency metrics and associated algorithms that can be used to automatically identify the dependencies between ISM practices.

For the identification of ISM practice dependencies, we consider the mappings between ISMs and ICM. Furthermore, we consider ISM practiceConcepts pc_{ISM_t} that are ISM explicit or implicit artifacts of type $t \in T$, $T = \{\text{ISM input, ISM output}\}$.

5.6.1 Dependency Metrics

According to the measurement theory, we define the measurement scale. The dependency metrics use an ordinal scale with the order: “Strong” > “Medium”> “Absent”.

Furthermore, the dependency metrics are not symmetric. For example DepPRACT ($pract_{ISM1}$, $pract_{ISM2}$) ≠ DepPRACT ($pract_{ISM2}$, $pract_{ISM1}$).

According to our design principle, we define the dependency metrics on different levels (Fig. 37).

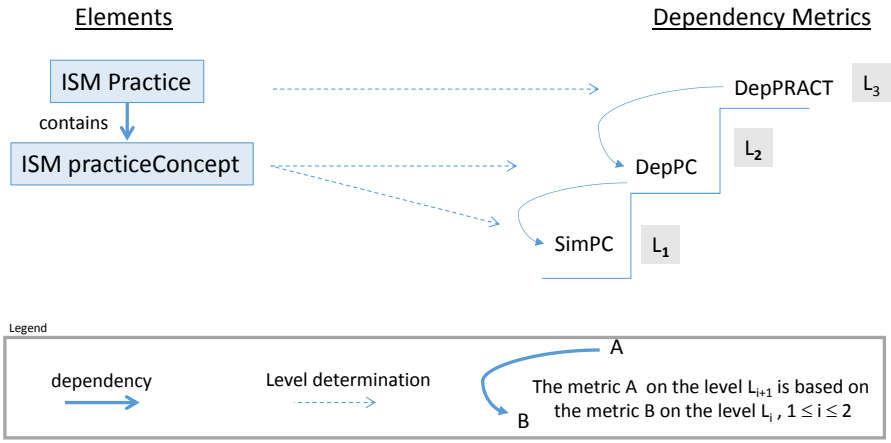


Fig. 37: Dependency metrics – Design

Firstly, each dependency metric depends on the dependency metric on the previous level. On the last level, it depends on the similarity metrics. The dependency metric on the ISM practice level depends on the dependency metric on the ISM practiceConcept level, that depends on the similarity metric on the ISM practiceConcept level.

Secondly, the dependency degree on a level is high on the given scale if the dependency or similarity degree on the previous level is also high on the given scale. For example, the dependency degree between two ISM practices is ”Strong” on the ordinal scale, if the dependency degree of their ISM practiceConcepts is ”Strong” on the ordinal scale. Their dependency degree of the ISM practiceConcepts is “Strong” on the ordinal scale, if their similarity degree is high on the ratio scale.

Finally, the dependency degree between ISM practices represents the final result.

5.6.1.1 Dependency Metrics for ISM PracticeConcepts

For a pc_{ISM_input} and a pc_{ISM_output}, the metric **DepPC (Dependency of Practice Concepts)** determines their dependency degree.

DepPC depends on the metric SimPC. If the ISM practiceConcepts are semantically equal (SimPC value = 1), then the dependency degree is “Strong”. If the ISM practiceConcepts are similar (0 < SimPC value < 1), then the dependency degree is “Medium”. Otherwise, the dependency degree is “Absent”. The metric is defined as follows:

$$\text{Dep}PC(pc_{ISM\_input}, pc_{ISM\_output}) = \begin{cases} \textit{„Strong“}, & \text{iff } SimPC(pc_{ISM\_input}, pc_{ISM\_output}) = 1. \\ \textit{„Medium“}, & \text{iff } 0 < SimPC(pc_{ISM\_input}, pc_{ISM\_output}) < 1. \\ \textit{„Absent“}, & \text{otherwise.} \end{cases}$$

5.6.1.2 Dependency Metrics for ISM Practices

For two practices $pract_{ISM1}$ and $pract_{ISM2}$, the metric **DepPRACT (Dependency for Practices)** determines their dependency degree.

We define the sets:

- $PC_{practISM\_input} = \{pc_{ISM\_input} \mid pc_{ISM\_input} \text{ is contained in } pract_{ISM2}\}$.
  This set contains all ISM inputs contained in $pract_{ISM2}$.
- $PC_{practISM\_output} = \{pc_{ISM\_output} \mid pc_{ISM\_output} \text{ is contained in } pract_{ISM1}\}$.
  This set contains all ISM outputs contained in $pract_{ISM1}$.

DepPRACT depends on the metric DepPC. The dependency degree between $pract_{ISM1}$ and $pract_{ISM2}$ is “Strong”, if the $pract_{ISM2}$ contains an ISM input that is produced as ISM output by the $pract_{ISM1}$ and the ISM input and output are semantically equal. If the ISM input and output are only similar, than the dependency degree is “Medium”. The metric is defined as follows:

$$\text{Dep}PRACT(pract_{ISM1}, pract_{ISM2}) = \begin{cases} \textit{„Strong“}, & \text{iff } \exists\, pc_{ISM\_input} \in PC_{practISM\_input} \text{ and } \exists\, pc_{ISM\_output} \in PC_{practISM\_output}, \\ & DepPC(pc_{ISM\_input}, pc_{ISM\_output}) = \text{„Strong“}. \\ \textit{„Medium“}, & \text{iff } \exists\, pc_{ISM\_input} \in PC_{practISM\_input} \text{ and } \exists\, pc_{ISM\_output} \in PC_{practISM\_output}, \\ & DepPC(pc_{ISM\_input}, pc_{ISM\_output}) = \text{„Medium“}. \\ \textit{„ Absent“}, & \text{otherwise.} \end{cases}$$

A dependency between two ISM practices with the dependency degree „Strong“ or „Medium“is called “Strong” and „Medium“ ISM practice dependency respectively.

5.6.1.3 Verification of Requirements

We only verify the achievement of the general requirement (R-All) as there are no specific requirements. This requests that the metrics' results should be differentiable, comparable, reproducible and plausible. As the dependency metrics are based on the similarity metrics that fulfill this general requirement, the results are differentiable, comparable and reproducible. The plausibility of the dependencies is verified during an experiment (section 8.3).

5.6.2 Algorithm

The algorithm identifies ISM practice dependencies. It considers as input an ISM practice $pract_{ISM}$ and delivers as output the ISM practices that are dependent on $pract_{ISM}$ with their corresponding DepPRACT values. It consists of the following steps:

1. Let DEP_PRACT = {}.

2. Identify PCpractISM_output = {pcISM_output | pcISM_output is contained in practISM}.

3. For each pcISM_output ∈ PCpractISM_output do:

3.1. Identify CONCpcISM = {concICM | concICM is related to pcISM_output}.
Identify CONCancestor = {concICM_ancestor | concICM_ancestor is an ICM ancestor concept of concICM ∈ CONCpcISM}.
Let CONCpcISM = CONCpcISM ∪ CONCancestor.

3.2. For each concICM ∈ CONCpcISM identify PCpractISM_input = {pcISM_input | pcISM_input is related to concICM}.

3.3. For each pcISM_input ∈ PCpractISM_input do:

3.3.1. Identify PRACTpcISM = {practISM | practISM contains pcISM_input}.

3.3.2. For each practISMi ∈ PRACTpcISM do:

3.3.2.1. Identify PCpractISM_input = {pcISM_input | pcISM_input is contained in practISMi}.

3.3.2.2. For each pcISM_input ∈ PCpractISM_input compute the DepPC(pcISM_input,pcISM_output).

3.3.2.3. Compute DepPRACT(practISM,practISMi).

3.3.2.4. If DepPRACT(practISM,practISMi) ≠ "Absent" then DEP_PRACT = DEP_PRACT ∪ {(practISM)}.

4. Return DEP_PRACT.

5.6.3 Example

We identify the dependencies for the ISM practice CMMI-DEV PPQA SP1.2.5 considering the ISMs and ICM illustrated in Fig. 38. The ISMs contains the following ISM practices:

- CMMI-DEV PPQA SP1.2.5 "Identify each case of noncompliance found during evaluations."
- SPICE SUP.1 BP9 "Ensure resolution on non-conformances."

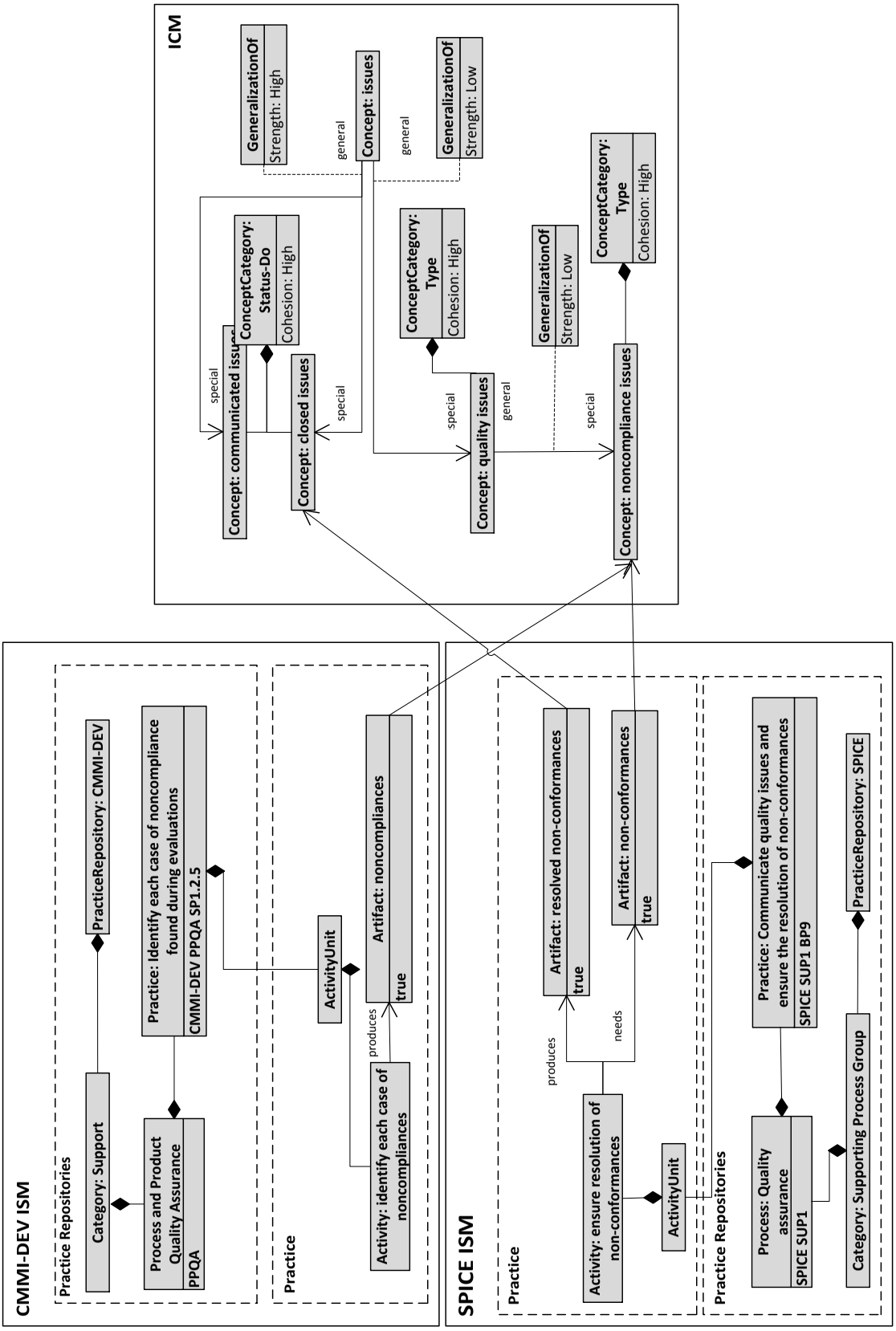


Fig. 38: Example – Identification of dependencies. Excerpts of ISMs and ICM

Table 19 illustrates the steps performed to compute the dependency degrees on the different levels.

Input		Step 1	Step 2	Step 3									
				Step 3.1	Step 3.2	Step 3.3							
						Step 3.3.1	Step 3.3.2						
							Step 3.3.2.1	Step 3.3.2.2	Step 3.3.2.3	Step 3.3.2.4			
ISM practice		DEP_PRACT	ISM practice elements ($PCpractISM_{output}$)	ICM concepts ($CONC_{pciSM}$)	ISM practice elements ($PCpractISM_{input}$)	ISM Practices ($PRACT_{pciSM}$)	ISM practiceConcepts in $PRACT_{pciSM}$	DepPC	DepPRACT	DEP_PRACT			
CMMI-DEV PPQA SP1.2.5		{}	{noncompliance issues}	{noncompliance issues, issues}	{noncompliance issues}	{SPICE SUP.1.BP9}	{non-conformances}	Strong	Strong	(SPICE SUP.1.BP9, Strong)			

Table 19: Example – Dependencies Algorithm

5.7 Summary

We defined a set of analysis activities that can be used by an Analyzer to effectively and efficiently deal with multiple PRs:

- Selection of ISM practices based on SFM situationalFactors
- Identification of similar ISM practices based on:
  - Similarity degree between ISM practices
  - Coverage degree of two sets of ISM practices (inclusively highest degree and the best coverage in a set of ISM practices)
  - Output states of ISM practices
- Identification of dependencies between ISM practices

These activities are realized by different algorithms and underlying metrics.

For each metric, we defined requirements and assumptions that are needed for their development. Furthermore, the metrics are defined on different levels that are determined by the elements and their relations in the IS and IC Meta-Models. Each metric on a level depends on the metric on the previous level, and each metric result on a level is high on the given scale if the metric result on the previous level is high on the given scale. The metric result on the first level represents the final result.

Based on the developed metrics, we illustrated several algorithms that implement the analysis activities. For a better understanding, we exemplified these algorithms as well, by using ISM practices from our example scenario related to the application quality. The defined algorithms are only examples of how to apply the metrics on the built models. Other algorithms are possible. For example, the similarity metrics can be applied to define a mapping between two ISM processes by identifying the pairs of ISM practices that have the highest similarity degree.

6 MOSAIC Toolbox

The MOSAIC Toolbox supports a Modeler to perform the modeling activities and an Analyzer to perform the analysis activities on the built models. It contains various tools that are integrated into a web application based on the multilayered standard architecture of the Java Platform Enterprise Edition (Java EE).

The MOSAIC Toolbox is developed in collaboration with two students and based on the results obtained during their master thesis at our research department [Dragomir 2011; Mageramov 2013].

In the following, we define the MOSAIC Toolbox requirements. Functional and non-functional requirements are generally defined for a software application. As the MOSAIC Toolbox aims to be a proof-of-concept, we do not define any non-functional, but only functional requirements. Furthermore, we present the MOSAIC Toolbox architecture and describe in more details one of its tools.

6.1 Functional Requirements

We use an UML use case diagram to illustrate the functional requirements of the MOSAIC Toolbox (Fig. 39).

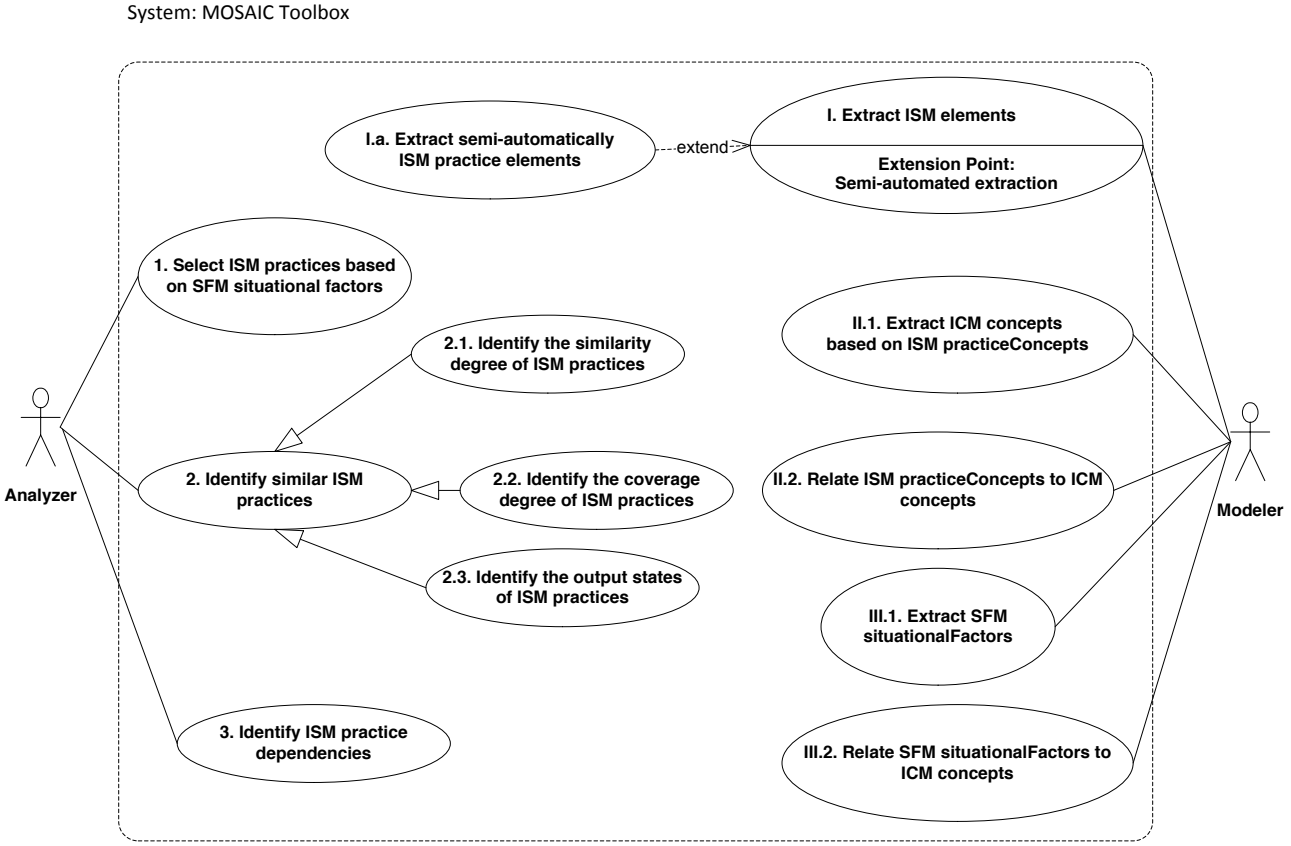


Fig. 39: MOSAIC Toolbox - Use Case Diagram

The use cases describe the interaction between an actor and the MOSAIC Toolbox.

The Modeler uses the system to perform the modeling activities. Therefore, the Modeler uses the system to extract the ISM elements from the PRs. For this extraction, he also has the possibility to semi-automatically extract the ISM practice elements. Based on the extracted ISM practiceConcepts, he extracts the ICM concepts and relates them. Finally, he extracts the SFM situationalFactors and relates them to the ICM concepts.

Based on the created models, the Analyzer can perform various analysis activities. He can select ISM practices based on SFM situationalFactors that characterize existing and possible critical situations in a software project. Furthermore, he can identify similar ISM practices. There are three possibilities to compare practices: he can identify the similarity degree, coverage degree or the output states of ISM practices. Finally, he can identify the dependencies between ISM practices.

6.2 Overview of the Tools within the MOSAIC Toolbox

The MOSAIC Toolbox refers to a set of tools that implement the requirements described in the last section. Fig. 40 gives an overview of these tools and a mapping between these tools and the implemented requirements.

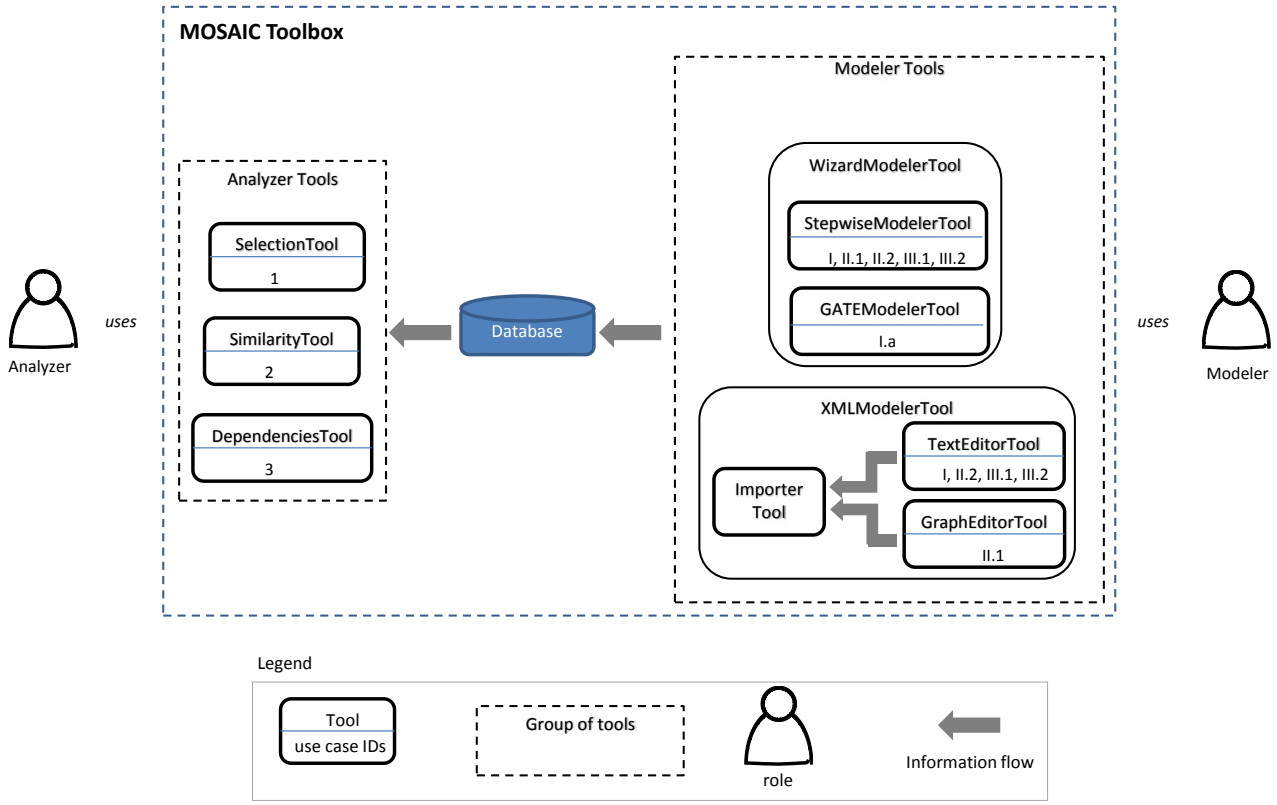


Fig. 40: MOSAIC Toolbox and corresponding use cases

An Analyzer uses the analyzer tools to perform the analysis activities on the data in the database. He can use the `SelectionTool`, `SimilarityTool` and the `DependenciesTool` to select ISM practices from multiple PRs based on the SFM situationalFactors, to identify similar ISM practices, as well as to identify dependencies between ISM practices of multiple PRs.

A Modeler uses different modeler tools to model the data in the database. We differentiate between the `WizardModelerTool` which guides the Modeler to perform the modeling activities and `XMLModelerTool` which can be used to model the data in XML format based on the MOSAIC metamodels.

The `WizardModelerTool` contains the following tools:

- The `StepwiseModelerTool` supports the Modeler step by step to manually perform all modeling activities.
- The `GATEModelerTool` supports the Modeler to semi-automatically extract the ISM practice elements. It uses GATE[10], a natural language processing-based application for this purpose.

The `XMLModelerTool` contains the following tools:

- The `TextEditorTool` supports the Modeler to perform all modeling activities with the exception of the extraction of ICM concepts.
- The `GraphEditorTool` supports the Modeler to extract the ICM concepts based on the ISM practiceConcepts.
- The `ImporterTool` reads the data in XML format, modeled in the `TextEditorTool` and `GraphEditorTool` and imports it in the database.

To summarize, Table 20 gives an overview of the different modeler tools and their corresponding use cases.

Use Cases	StepwiseModelerTool	GATEModelerTool	TextEditorTool	GraphEditorTool	ImporterTool	Goal
Extract ISM elements	✓	✓	✓	-		Structure normalization of PRs
Extract manually ISM practice elements	✓	-	✓	-		
Extract semi-automatically ISM practice elements	-	✓	-	-		
Extract ICM concepts based on ISM practiceConcepts	✓	-	-	✓		Terminology normalization of PRs
Relate ICM concepts to ISM practiceConcepts	✓	-	✓	-		
Extract SFM situationalFactors	✓	-	✓	-		Modeling of the Software Project Context and Integration with the PRs
Relate SFM situationalFactors to ICM concepts	✓	-	✓	-		

Table 20: Use cases and corresponding modeler tools

[10] GATE at http://gate.ac.uk/

6.3 Architecture

In this section, we describe the architecture of the web application that integrates the components of the MOSAIC Toolbox tools.

The architecture does not contain components for the `TextEditorTool` and `GraphEditorTool`. A Modeler can use Notepad or other text editor as a `TextEditorTool` and the yED Graph Editor[11] as a `GraphEditorTool` to model the data in XML format. Therefore, we did not implement these tools by our own. The architecture description only contains the self-developed components.

6.3.1 Overview

The architecture is based on the multilayered standard architecture of the Java Platform Enterprise Edition (Java EE). Hence, the MOSAIC Toolbox architecture also contains two tiers that run on the Java EE Server: *web* and *business tier*. We differentiate between web and business components accordingly (Fig. 41):

- The *web components* are the `WebModeler` and `WebAnalyzer`. These interact with the Modeler and the Analyzer who aim to perform the modeling and analysis activities respectively. Each contains components that correspond to the MOSAIC Toolbox tools (e.g. `WebWizardModeler` or `WebImporter`).
- The *business components* are the `BusinessModeler` and `BusinessAnalyzer`. These are defined to offer a single facade for the web components. The `BusinessModeler` is used by the web and business components to get access to the data in the database. The `BusinessAnalyzer` contains the components that implement the logic of the analysis activities and is used by the `WebAnalyzer`.

[11] yED Graph Edit at http://www.yworks.com

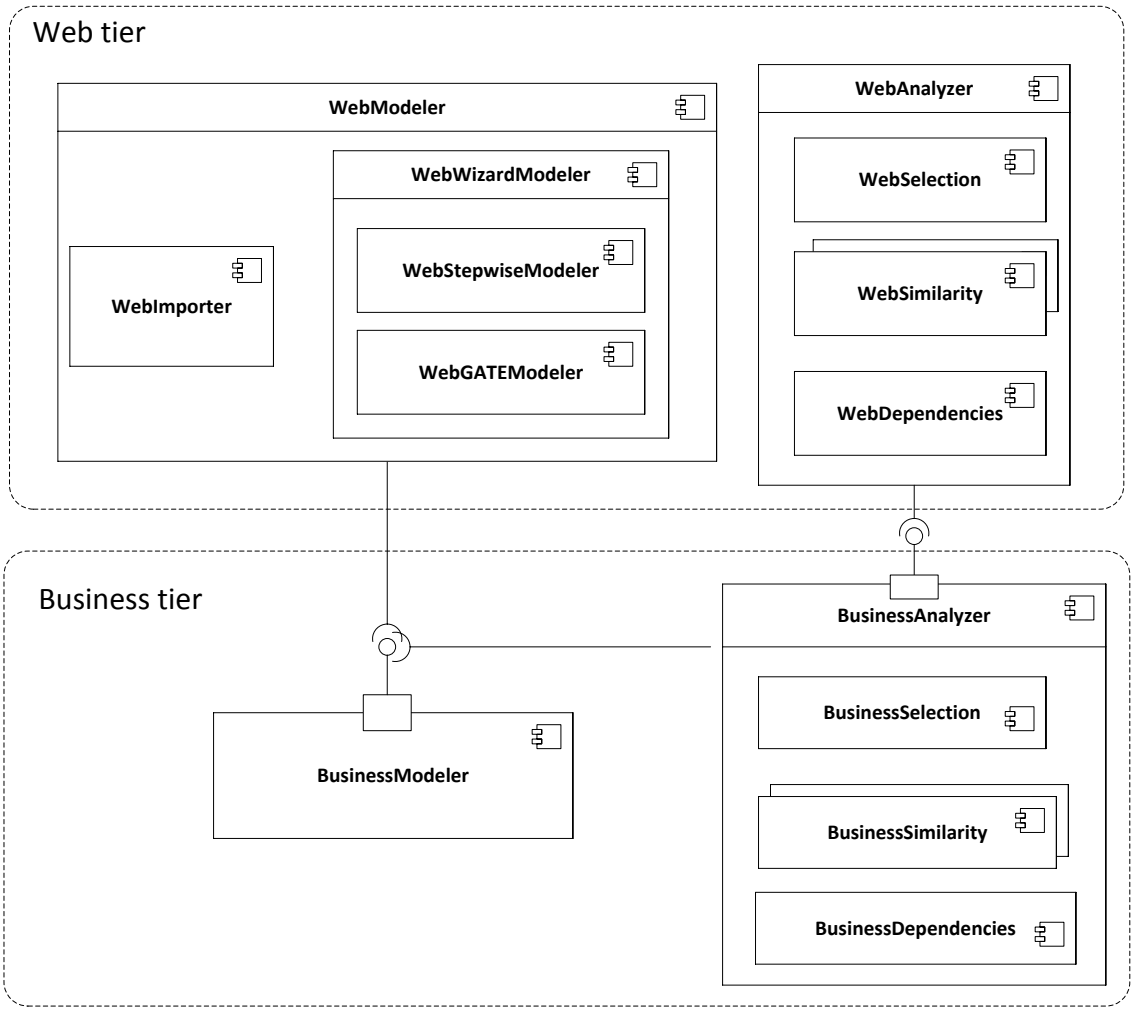


Fig. 41: MOSAIC Toolbox architecture

The MOSAIC architecture is loosely coupled and defines dedicated components for each of the tools within the MOSAIC Toolbox. Consequently, the components of our architecture can be extracted and integrated into a single application based on the needs of an organization (Table 21). For example, if an organization is only interested to create a repository of multiple PRs with a common structure, then the components of one of the modeler tools, e.g. the `WizardModelerTool`, can be extracted and integrated into an application. If an organization is only interested in the selection of ISM practices based on the software project context, then the components of the `SelectionTool` can also be extracted and integrated to a single application. The `BusinessModeler` component has to be used by all the tools as it is the component that interacts with the database and provides access to the data.

Table 21 gives a mapping between the architecture components and the MOSAIC Toolbox tools.

Components/Tools	Modeler Tools		Analyzer Tools		
	StepwiseModelerTool	XMLModelerTool	SelectionTool	SimilarityTool	DependeciesTool
WebImporter		✓			
WebStepwiseModeler	✓				
WebGATEModeler	✓				
BusinessModeler	✓	✓	✓	✓	✓
WebSelection			✓		
WebSimilarity				✓	
WebDependencies					✓
BusinessSelection			✓		
BusinessSimilarity				✓	
BusinessDependencies					✓

Table 21: Mapping between the architecture components and MOSAIC Toolbox tools

6.3.2 Web Tier

Fig. 42 exemplifies the detailed architecture of the web components. Based on the Java Server Faces (JSF) framework, there are different components of the `WebModeler` and `WebAnalyzer`:

- `Facelets` realize the user interface that is run by the web browser.
- `Managed Beans` handle an event fired by the `Facelets`, validate the data received from the user and perform the processing to call the business components. In each `WebModeler` component, the `ManagedBeans: Presenters` use the `ManagedBeans: PresenterHandlers` to validate the user request before sending the create, update or delete request to the business components. In the `WebAnalyzer` components, there is no need to validate the user request as the input data for each analysis activity is predefined in the user interface. Therefore, the call is directly delegated to the business components by the `ManagedBeans: Presenters`.

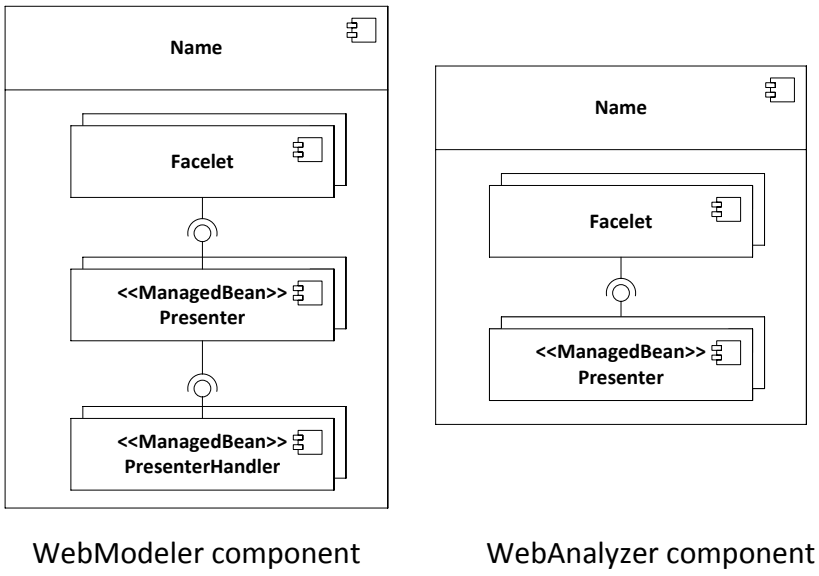


Fig. 42: Detailed Architecture of the web components

6.3.3 Business Tier

The business components in the *business tier* are used by the web components to process or get the data from the database. The lowest level components are the EJBs (Java Enterprise Beans). These are standard components within the Java EE framework. Fig. 43 illustrates the detailed architecture of the business components.

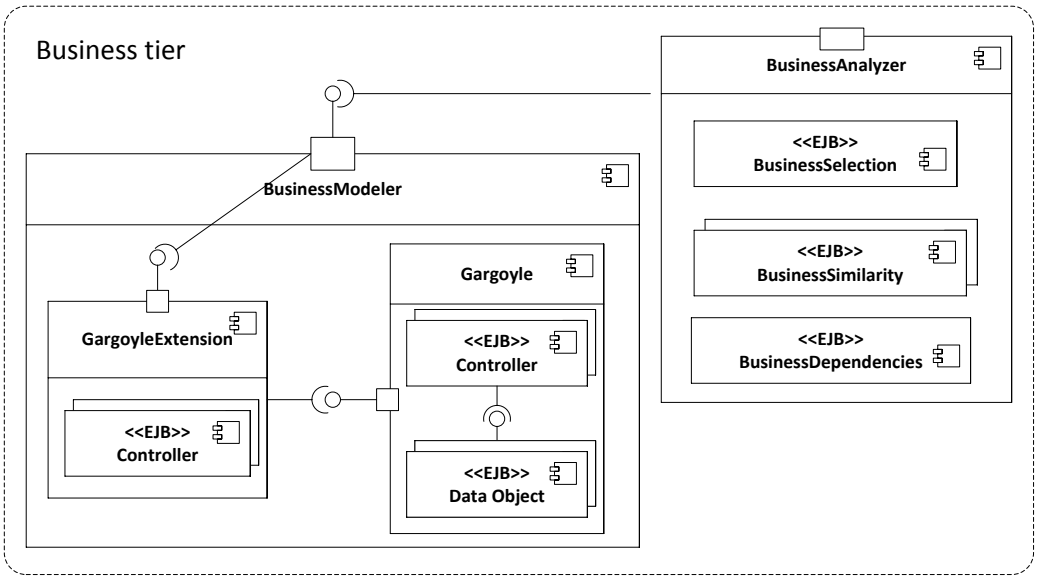


Fig. 43: Detailed architecture of the business components

The `BusinessModeler` components are used by the `WebModeler` components to create, update, remove or get the data into or from the database. It is mainly based on the components generated by `Gargoyle`, a model driven software development tool developed at our department [Löwenthal 2011]. `Gargoyle` generates various EJBs based on meta-models. We used this tool to generate the

`Gargoyle` component with its `Controller` and `Data Object` components based on the MOSAIC meta-models. This generation allows us to easily maintain the `BusinessModeler` when the MOSAIC meta-models change. For each meta-model element, the `Gargoyle` tool generated the corresponding `Data Object` and `Controller` components. The `Data Objects` create and interact with the database. The `Controllers` control the creation, update or removal of these `Data Objects` by verifying that certain conditions are fulfilled. As the `Gargoyle` component did not always fit our purposes, we developed the `GargoyleExtension` component which wraps and extends the Gargoyle component functionality:

- It extends methods if the `Gargoyle` component does not consider all needed aspects (e.g. deletion of an ICM concept must consider also the deletion of its corresponding similarity relations)
- It implements new methods (e.g. gets all ISM practices for an ISM process from the database for the identification of ISM practice dependencies)

The `BusinessAnalyzer` is used by the `WebAnalyzer` components to call the various implementations of the analysis activities. These are implemented by the `BusinessSelection`, `BusinessSimilarity` and `BusinessDependencies`. Furthermore, the `WebAnalyzer` uses the `WebModeler` component to interact with the needed data.

6.3.4 Analysis of the Architecture

We argue that the MOSAIC architecture is simple and flexible because of the following reasons:

- *Dedicated components*. The components of our architecture can be combined and integrated to build single applications that are needed for an organization. Furthermore, modifications of a certain functionality only affect the components that implement it.
- *Extension of analysis components*. Different analysis activities are implemented by different analysis components and thus, the integration of new analysis activities can be easily performed.
- *Modifications of modeling components*. We used the `Gargoyle` tool to generate the modeling business component that interact with the database. This modeling component can be easily replaced by another component, i.e. a new `Gargoyle` generated component or a component generated by other tools. Consequently, modifications in the MOSAIC meta-models do not require complex modifications.
- *Multilayer pattern*. The communication between the components is based on the multilayered standard Java EE architecture and thus, the communication is being kept simple.

Furthermore, to evaluate the simplicity and flexibility of the MOSAIC Toolbox architecture, we performed a static analysis with the Sonargraph[12]. Key metrics such as overall coupling and the level of package cyclicity are indicators for a simple and flexible architecture of a software (Table 22). We analyzed the results based on the interpretation guidelines by Sonargraph[13].

[12] Sonograph available at https://www.hello2morrow.com/products/sonargraph
[13] Definition for Sonargraph metrics available at https://www.hello2morrow.com/products/sonargraph/sonar

Tool	Parameter	Resulted Value	Definition and Interpretation
Sonargraph	Relative Package Cyclicity	26.32%	The metric gives an indication, how cyclic is the system on the package level. The resulted value indicates a warning, as the underlying package cyclicity is more than 25% but less then 50%.
	Cyclic packages	7.00	The metric gives an indication of the total number of packages involved in cyclic package dependencies. The resulted value is not high according to the interpretation guidelines provided by Sonargraph.
	Biggest Cycle Group Size	4.00	The metric is the size of the biggest cycle group. The resulted value 4 indicates a warning.
	Type Dependencies to Cut	45.00	The first metric indicates how many type dependencies need to be cut to break up all package cycles. The second metric indicates how many lines in the program code would be affected by this change. The resulted values is not high as the number of LOC for the java files is 55770 Therefore, approximately 1% of the code is affected.
	Type References to Remove	507.00	
	Highest ACD	15.63%	This metric indicates the overall coupling. As the resulted value is smaller than 50%, the overall coupling is not high.
	Highest NCCD	2.26%	This metric is the normalized version of ACD, that is independent of the system size. The resulted value is not high and does not indicate a warning as it is less than 6.5.

Table 22: Sonargraph code metrics

The afore mentioned arguments indicate that the MOSAIC architecture is simple and flexible.

6.4 Implementation

The MOSAIC Toolbox is implemented according to its architecture. We shortly describe an analysis that we performed to sustain this fact. Furthermore, we give a list of code metrics that give information about the size of the MOSAIC Toolbox. Finally, as the implementation of the `GATEModelerTool` is a complex component of the MOSAIC Toolbox, we describe it separately.

The appendices contain a short handbook of the MOSAIC Toolbox (chapter 10.2). Here, we give guidelines how to use the tools within the MOSAIC Toolbox.

6.4.1 Architecture Conformance

The MOSAIC Toolbox is conformant to its architecture. To verify this, we performed a dynamic analysis with the CIC (Communication Integrity Checker), a tool developed within the ARAMIS project in our research department [Dragomir and Lichter 2013]. The main goal of the ARAMIS project is to support organizations in the development of systems according to their architecture. It ensures that implemented use cases do not violate predefined architectural rules. Therefore, we could verify if the communication inside the MOSAIC Toolbox is performed only in ways intended by its architecture.

Together with a master student, we defined different scenarios of using the MOSAIC Toolbox. For example, we extracted ISM practices and its practice elements from CMMI-SVC. While we performed these modeling activities, the CIC registered the communication inside the MOSAIC Toolbox. Based on an analysis of the CIC results, we discovered that CIC identifies 3 kinds of violations with a total of 25 frequency within the used scenarios in more than one hundred thousand checked communication attempts (Table 23).

Calling Entity	Called Entity	Frequency
WebModeler: Presenters	BusinessAnalyzer: AnalysisFacade	2
WebModeler: Presenters	BusinessModeler: Gargoyle: FacadeExtension	1
WebModeler: Presenters	BusinessModeler: ModelingFacade	22

Table 23: Types of architectural rules violations

For example, one type of violation is that the `Presenters` of the `WebModeler` web components use the analysis business components (`AnalysisFacade`). Another type is that the `Presenters` of the `WebModeler` directly interact with the business components (`FacadeExtension` and `ModelingFacade`) and not via the `PresenterHandlers` of the `WebModeler`. Such behaviors are not allowed according to the defined architecture.

To summarize, we can argue, that the MOSAIC Toolbox is developed according to its architecture.

6.4.2 Code Metrics

In this section, we give some information about the size of the MOSAIC Toolbox implementation. Table 24 lists some key metrics computed in by Code-Stats[14] which give information about the size of the MOSAIC Toolbox software system.

	Source files	Lines of Code
Code Stats	Java	55770
	XML	3977
	JS	34570
	CSS	8050
	XHTML/HTML	2462
	JAPE Transducers	2138
	Others	4786
	Total Lines of codes	**111753**

Table 24: MOSAIC Toolbox size metrics

All the source files except for the JAPE Transducers are parts of the Java web application. The JAPE (Java Annotation Patterns Engine) Transducers files are parts of the GATE application, that we implemented to automatically extract ISM practice elements from an ISM practice. The Java web application uses the output of these JAPE Transducers to finalize the extraction of the ISM practice elements. This process and thus, the interface between the GATE and Java web application is implemented in the `GATEModelerTool`. Therefore, `GATEModelerTool` automatically extracts the ISM practice elements. We describe this tool in the following.

[14] Code-Stats available at https://npmjs.org/package/code-stats

6.4.3 GATEModelerTool

In this section, we describe one of the tools within the MOSAIC Toolbox, namely the `GATEModelerTool`, as its implementation based on the natural processing language techniques is not straightforward and need some explanations.

The `GATEModelerTool` extracts ISM practice elements based on the identification of grammatical structures in the text of an ISM practice. The extraction of ISM practice elements means not only their identification, but also a normalization of their language. According to their personality, educational and cultural background the authors of PRs tend to express the same ideas differently. For example, while in COBIT and Functional Safety practices are abundant in passive structures, in CMMI practices are written almost completely using the active form. A normalized writing style, e.g. using only active structures, supports a better understanding of the PRs and leads to a consistent modeling of its elements in MOSAIC.

We developed this tool iteratively and improved the results after each iteration. The architecture and results of the first phase are described here [Jeners et al. 2012a]. The architecture of the improved tool is described in the next sections.

6.4.3.1 Overview

Based on various Natural Language Processing (NLP) tasks and on the tool support offered by GATE (General Architecture for Text Engineering) [Cunningham et al. 2012] and the Dragon Toolkit [Zhou et al. 2007], we constructed the `GATEModelerTool`. This parses an ISM practice, normalizes and forms the ISM practice elements based on various extraction rules. We define such rules based on language structure guidelines (normalization rules) and based on the elements and their relations defined by the IS Meta-Model (forming rules) (Fig. 45).

To specify these rules we use the sign "→" to express that the left side is transformed into the right side. The left and right side are specified using a variation of EBNF (Extended Backus–Naur Form). We use the sign "+" instead of "," because in linguistics "+" is often used to concatenate grammatical structures. Moreover, when a lexical structure has to have a certain value for a rule to apply, then the allowed values are written under the rule and are separated by the sign "|". For example, Fig. 44 illustrates such a rule, where the accepted values for the preposition are listed. This rule transforms a verb that can be preceded by an adverb and followed by a noun, a possible relative sentence, preposition and a noun into a verb followed by the first noun.

```
{Adverb} + Verb + Noun1 + {RelativeSentence} + {Preposition+Noun2} → Verb + Noun1

Preposition = "by" | "for" | "on which" | "which are in" | "with" | "amongst" | "for" | "in" | "inside" | "that" | "in terms of" | ..
```

Fig. 44: GATEModelerTool – Example of a rule

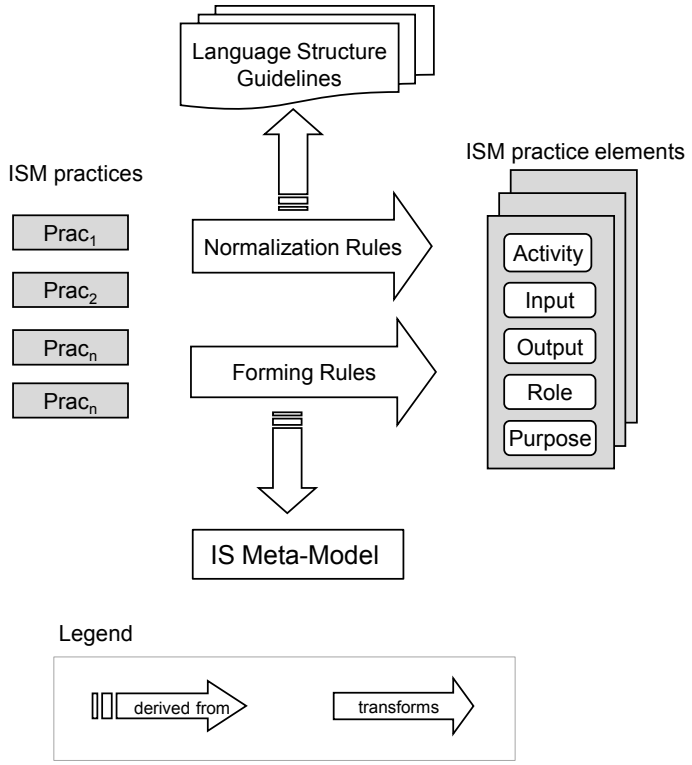


Fig. 45: Normalization and forming of ISM practice elements

The `GATEModelerTool` contains a GATE application and a Java application with different components which communicate with each other in a chain according to the pipes and filters architecture style – one component uses as input the output of the previous component (Fig. 46).

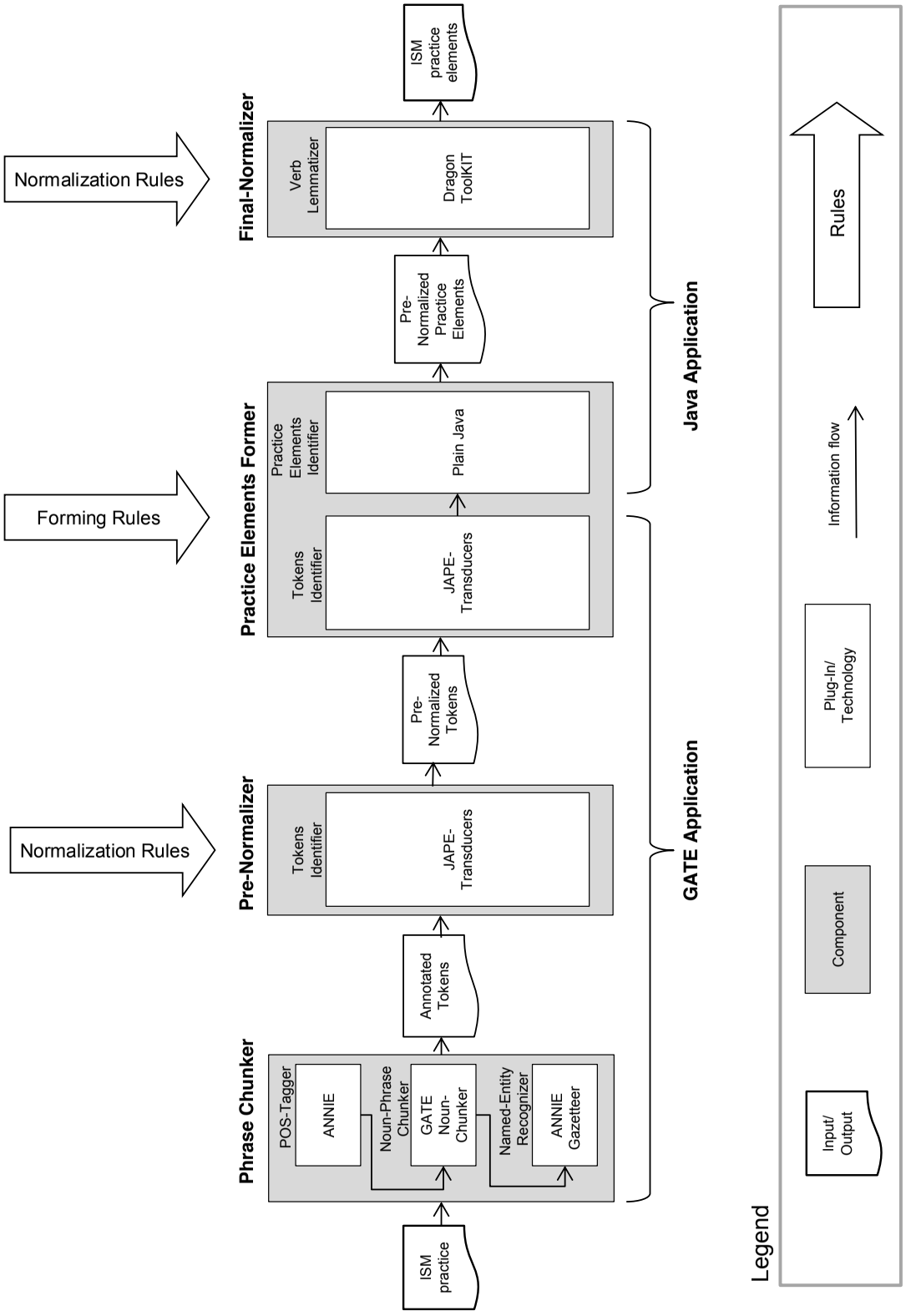


Fig. 46: Extraction of ISM practice elements components chain

Firstly, we built the GATE application in the GATE environment that enables the developer to run and test applications based on NLP tasks. These NLP tasks are performed by various plug-ins within the GATE application. Shortly, the GATE application consists of three components:

- `Phrase Chunker` chunks the sentences in tokens and annotate them accordingly.
- `Pre-Normalizer` uses the normalization rules to perform a first normalization of the annotated tokens.
- A part of the `Practice Elements Former`, namely the `Tokens Identifier`, annotates the pre-normalized tokens as activities, outputs, inputs, roles and purposes tokens based on the forming rules.

Secondly, we instantiated the GATE application in the Java application that extracts the ISM practice elements based on the token annotations from the GATE application and finalizes the normalization based on the normalization rules. It consists of two components:

- A part of the `Practice Elements Former`, namely the `Practice Elements Identifier`, identifies the ISM practice elements based on the GATE application annotations.
- `Final-Normalizer` with its `Verb Lemmatizer` finalizes the normalizations by transforming passive or gerund form of verbs in the active form.

Table 25 illustrates the extraction of ISM practice elements from an ISM practice. We modified the original form of the ISM practice CMMI-DEV PPQA SP2.1 to illustrate the normalization of the passive form of the verb. The original text of this ISM practice does not contain passive structures. First, the `Phrase Chunker` annotates the tokens as verbs and nouns with the support of its three plug-ins: `ANNIE`, `Noun-Phrase Chunker` and `Named-Entity-Recognizer`. Furthermore, the `Pre-Normalizer` transforms the verb token and extracts the auxiliary form of the verb. The `Practice Elements Former` forms and annotates the pre-normalized tokens as activities, outputs and roles and then forms the ISM activities, outputs, inputs and roles accordingly. The ISM outputs and inputs are formed based on the output and activity annotations. Finally, the `Final-Normalizer` transforms the extracted verb from its passive to its active form.

Table 25: Example – Forming and normalization ISM practice elements by the tools' chain

ISM practice	Phrase Chunker and Plugins	Pre-Normalizer	Practice Elements Former Tokens Identifier	Practice Elements Former Practice Elements Identifier	Final-Normalizer
	Forming and Annotation	Transformation	Forming and Annotation	Forming and Annotation	Transformation
Modified CMMI-DEV PPQA SP2.1 Quality issues are communicated and noncompliance issues are resolved with the staff and managers.	**Verb**: are communicated, are resolved (ANNIE)	**Verb:** communicated, resolved	**Activity:** communicated, resolved	**ISM activity:** communicated, resolved	**ISM activity**: communicate, resolve
	Noun: quality issues, noncompliance issues (Noun-Phrase Chunker)	-	**Output:** quality issues, noncompliance issues	**ISM output:** communicated quality issues, resolved noncompliance issues	-
		-	-	**ISM input:** quality issues, noncompliance issues	-
	Noun: staff, manager (Named-Entity Recognizer)	-	**Role:** staff, manager	**ISM role:** staff, manager	-

In the following, we describe in detail these components and describe the NLP tasks used.

6.4.3.2 Phrase Chunker

The `Phrase Chunker` splits a given ISM practice in Annotated Tokens based on various NLP tasks:

- *Part-of-Speech Tagging (POS Tagging)*, implemented by the ANNIE plug-in, identifies tokens (such as verbs, nouns, adjectives, adverbs) and annotates them correspondingly.
- *Noun-Phrase Chunking*, implemented by the GATE Noun-Chunker plug-in, annotates tokens as composed nouns.
- *Name-Entity Recognition*, implemented by the ANNIE-Gazetter plug-in, annotates tokens as nouns and supports the identification of the ISM roles.

Firstly, the *POS Tagging* chunks a sentence in POS tokens which are relevant for the identification of all ISM practice elements. POS tokens are "*a category to which a word is assigned in accordance with its syntactic functions*"[15]. The set of all POS tokens is different from language to language. For example, in English, the main POS categories are noun, pronoun, adjective, determiner, verb, adverb, preposition, conjunction and interjection. There exist two main POS tagging techniques: rule-based and probabilistic training POS tagging [Nau 2010]. On the one side, a rule-based POS tagging system uses a list of possible tagging sequences as rules and verifies which best suit the input sentence. On the other side, a probabilistic training system computes the occurring probabilities of the POS based on a previously annotated text corpora (e.g. Brown Corpus which contains over one million annotated words [Francis and Kucera 1979]). The probabilistic training system is often used as its results are more qualitative and the required storing space is less. However, the POS Tagging do not always provide the correct results due to language ambiguities. For example, the token “Estimate” from the practice CMMI-DEV PP SP1.4 “Estimate the project effort (…) based on estimation rationale” can be interpreted as a verb, but also as a noun.

We used the ANNIE (A Nearly-New Information Extraction System) plug-in [Cunningham et al. 2012] for the POS Tagging. There are various tool support for this NLP task (e.g. Stanford Parser [Marneffe et al. 2006], Dragon Toolkit [Zhou et al. 2007] or UIMA (Unstructured Information Management Architecture) [Ferrucci et al. 2006]). As ANNIE is a GATE plug-in, we used it for our purposes. Moreover, the results are satisfactory and the time performance is high. ANNIE uses a rule set, the result of a training on a large corpus taken from the Wall Street Journal, to identify the POS tokens.

Secondly, the *Noun Phrase Chunking* [Bird 2009] identifies noun phrases which are relevant to form the ISM artifacts and roles. A noun phrase is a combination of more nouns that have a standalone semantic meaning. For example, “work breakdown structure” is a noun phrase composed of the nouns “work”, “breakdown” and “structure”. As the Noun Phrase Chunking is based on the output of the POS Tagging task, it relies considerably on the precision of the used POS Tagger. There are various tool support for the Noun Phrase Chunking (e.g. Dragon Toolkit Xtract [Smadja 1993] or NounChunker [Cunningham et al. 2012]). As the NounChunker is a GATE plug-in, we used it to implement this NLP task. This GATE plug-in is a Java implementation of the Ramshaw and Marcus BaseNP chunker [Ramshaw and Marcus 1999].

[15] "Part Of Speech", Oxford Dictionaries http://oxforddictionaries.com/definition/part+of+speech? region=us

Finally, the *Named Entity Recognition* identifies specific nouns and supports the identification of ISM roles. It analyses a given text and identifies all the entities belonging to certain predefined categories. Such categories can be "persons", organizations", "locations", "roles (actors and stakeholders)". The ANNIE Gazetteer, also a GATE plug-in, identifies named entities based on some predefined list of entities.

6.4.3.3 Pre-Normalizer

Based on the annotated tokens, the `Pre-Normalizer` identifies the pre-normalized tokens. Based on the normalizations rules, such transformations are possible. There are various tools that are based on rules. For example, JAPE (Java Annotation Patterns Engine) Transducers, Prolog, SAIL (Semi-Automated Interactive Learning) or UIMA are tools to perform such transformations. The `Tokens Identifier` within the `Pre-Normalizer` utilizes the JAPE Transducers. The JAPE "provides a regular-expression based pattern/action rules over annotations" [Cunningham et al. 2012].

6.4.3.3.1 Normalization Rules

The `Pre-Normalizer` uses the JAPE Transducers to implement the normalization rules. It performs the first step in the normalization of the original text. The second and final step is performed later by the `Final-Normalizer`.

To define the normalization rules, we analyzed several writing style guidelines. First, we analyzed some recommendations from Requirements Engineering that are based on best practices from Neuro-Linguistic Programming [Grinder and Bandler 1976]. These recommendations are defined by the SOPHISTS to write clear, consistent, complete and unambiguous requirements [Pohl 2011], [Geltinger 2010]. Furthermore, we analyzed further guidelines for scientific writing[16]. Based on all these best practices, we recommend the application of the following rules.

- The **active form** of the verb has to be used instead of **passive voices**, **modal** and **present continuous tenses**.
  For example, instead of the passive structure in the COBIT PO 9.4 "The likelihood […] should be determined individually, by category and on a portfolio basis", the active form have to be used "Determine the likelihood […]".
- The corresponding verb in active form has to be used instead of **nominalizations** and **gerunds.**
  For example, instead of the nominalization in CMMI-DEV IPM SP 1.2 "Use organizational process assets and the measurement repository for estimating and planning project activities", the corresponding verbs in active form have to be used: "Use organizational process assets and the measurement repository to estimate and plan project activities". However, not all gerunds have to be transformed to their corresponding active form. For example in the practice CMMI-DEV IPM SP1.3.2 "Provide ongoing maintenance and operational support for the project's work environment.", the gerund "ongoing" is an adjective and can be left un-normalized.

The `GATEModelerTool` only implements the first rule (the active form of the verb has to be used instead of passive voices, modal and present continuous tenses). This is because the semantics

[16] Literature for Scientific Writing - http://abacus.bates.edu/~ganderso/biology/resources/writing/HTWgeneral.html, http://www.columbia.edu/cu/biology/ug/research/paper.html, http://maic.jmu.edu/journal/index/General_Rules_for_Scientific_Writing.pdf, http://www.ugr.es/~agcasco/tierra/Docs/kowalski_scientific_writing.pdf, http://faculty.uca.edu/march/bio1/sciwriting/writ-tips.htm.

is very important for the identification of the nominalizations and gerunds. An automated normalization would become too complex. Consequently, based on the first writing style guideline, we define normalization rules to transform the original forms in normalized forms (Table 26). Not all verb tenses need a two-step normalization. If the verb is in its active or in the modal form (rule 1 and 2), then no second step needs to be performed any more by the `Final-Normalizer`. The second step has to be performed only for the rule 3 and 4 and thus, it is marked accordingly (Table 26 – rule 3a and 4a).

Verb Tenses				
Id	**Normalization Rule**	**Description**	**Example - Original**	**Transformation**
1	Main verb in active → Main verb in active	The verb in the active form form remains in its active form.	ACQ.15.BP1: Establish criteria for qualifying suppliers	Establish
2	Verb in modal (modal auxiliary + main verb) → Main verb	The main verb is extracted.	COBIT PO10.7 The plan should include details of project deliverables	Include
3a	Verb in passive (auxiliary verb + main verb in past participle) → main verb in past participle	The main verb in past participle is extracted.	CMMI OT SG2: Training for individuals (...) is provided.	Provided
4a	Verb in present continousin active (auxiliary verb + -ing form of the main verb) → -ing form of the main verb	The -ing form of the main verb is extracted.	COBIT PO4.11 (..) personnel are performing only authorised duties.	Performing

Table 26: Normalization rules – First step

6.4.3.4 Practice Elements Former

Within the GATE application, the `Tokens Identifier` within the `Elements Former` `Practice Elements Former` uses the JAPE-Transducers to implement the forming rules and to annotate the pre-normalized tokens as activities, outputs, inputs, roles and purposes.

Within the Java application, the `Practice Elements Identifier` `Practice` within the `Elements Former` uses Plain Java to implement the forming rules and thus, form the ISM activities, outputs, inputs, roles and purposes elements based on the tokens received from the GATE application.

6.4.3.4.1 Forming Rules

The forming rules are based on the ISM practice language elements that define the lexical structure of the ISM practice elements.

1. Activity				
Id	**Forming Rule**	**Description**	**Example - Original**	**Annotation and Forming**
1	{Adverb} + Verb + Noun1 + {RelativeSentence} + {Preposition+Noun2} → Verb + Noun1 Preposition = "by" | "for" | "on which" | "which are in" | "with" | "amongst" | "for" | "in" | "inside" | "that" | "in terms of" | ..	A verb that can be preceded by an adverb and followed by a noun , relative sentences or nouns introduced by prepositions is annotated and then formed to an activity composed of the adverb, verb and its noun.	CMMI PP SP1.1 Establish a top level work breakdown structure (WBS) to estimate the scope of the project.	Activity: Establish a top level work breakdown structure (WBS)
2	{Adverb} + Verb + Preposition + RelativeSentence → Verb + Preposition + RelativeSentence Verb = "verify" |.. Preposition = "that" | "if" |..	A verb that can be preceded by an adverb and is followed by a preposition and relative sentence is annotated and then formed to an activity composed of the adverb, verb, preposition and the relative sentence.	COBIT PO7.2 (..)verify that personnel have the competencies	Activity: Verify that personnel have the competencies
3	Noun1 + Verb + Noun2 → "Establish" + Noun1 Verb="include" |.. Noun is not a Role	A verb introduced by a noun and followed by another noun is annotated as an activity composed of the verb "establish" and the noun before the verb.	COBIT PO10.7 The plan should include details of project deliverables (Include details of project	Activity: Establish a plan
4	Verb1 + Noun + Preposition + Verb2 + Noun2 → Verb2 + Noun2 Verb1 = "encourage" | "ensure" | make sure"|..	A verb followed by a noun, a preposition, a second verb and a second noun is annotated as an activity composed of the second verb and the second noun.	COBIT DS4.4 Encourage IT management to define change control procedures COBIT AI7.6 Ensure that the plan considers security	Activity: Define change control procedures Activity: Consider security

Table 27: Rules to form ISM activities

2. Purpose				
Id	**Forming Rule**	**Description**	**Example - Original**	**Annotation and Forming**
1	Verb + Noun + Relative Sentence → Relative Sentence	A relative sentence preceded by a verb and noun is annotated as a purpose.	CMMI PI SP 3.4.2. Use effective methods to package the assembled product.	Purpose: to package the assembled product
2	Verb + Noun1 + Preposition + Noun2 → Preposition + Noun2 Preposition = "by" |"for" | "on which" |	A preposition together with a noun preceded by a verb and another noun are annotated and then formed to a purpose.	SPICE SPL.2.BP13 The product is delivered to the intended customer with positive confirmation of receipt.	Purpose: with positive confirmation of receipt
3	Verb + Noun1 + Preposition + Noun2 → Preposition + Noun Preposition = "to"	A preposition "to" and a noun preceded by a verb and another noun are annotated and then formed to a purpose. This purpose will be transformed in a relative Sentence ("to" + Verb derived from Noun2) in a future step.	COBIT PO5.2: (..) maximise IT's contribution to optimising the return	Purpose: to optimising the return

Table 28: Rules to form ISM purposes

3. Input				
Id	**Forming Rule**	**Description**	**Example - Original**	**Annotation and Forming**
1	Verb + Noun1 + {Preposition} + Noun2 → Noun2 Preposition = "in line with"| "based on"| "commensurate with"| "against"| "required by"| "to address"| "according to"| "in the event of"| "in case of"| "introduced/incorporated/integrated into"|	A noun that can be introduced by a preposition and is preceded by a verb and another noun is annotated and then formed to an input.	COBIT AI5.3 Select suppliers according to a formal practice	Input: formal practice
2	Verb + Noun → Noun Verb ≠	A noun preceded by a verb that is not the mentioned in the list is annotated and then formed to an input.	CMMI-DEV IPM SP2.3: Communicate quality issues	Input: quality issues
4. Output				
Id	**Forming Rule**	**Description**	**Example - Original**	**Annotation and Forming**
1	Verb + Noun → Noun Verb =	A noun preceded by a verb that is mentioned in the list is annotated and then formed to an output.	SPICE ACQ.15.BP1: Establish criteria for qualifying suppliers	Output: criteria for qualifying suppliers
2	Verb + Noun → Adjective (derived from verb) + Noun Verb ≠ "establish"|"create"|"define"|"make"|"ensure"	A noun preceded by a verb that is not mentioned in the list is annotated and then formed to an output composed of an adjective derived from the verb and the noun.	CMMI-DEV IPM SP2.3: Communicate quality issues	Output: communicated quality issues

Table 29: Rules to form ISM artifacts

5. Role				
Id	**Forming Rule**	**Description**	**Example - Original**	**Annotation and Forming**
1	Verb + Preposition + Noun → Noun Preposition = "with"| "for"| "of"| "amongst all"| "by"| "in cooperation with"| "from"| "between"| "to| "amongst" | .. Verb= "report to" | "approve by" | "is transparent to" | "involve" | "drive" |	A noun introduced by a preposition and preceded by a verb is annotated and then formed to role.	COBIT PO4.8 Obtain direction from senior management	Role: senior management
2	Noun1 + Verb + Noun2 → Noun1	A noun followed by a verb and another noun is annotated and then formed to role.	COBIT PO4.9 Owners should make decisions about classifying information	Role: owners
3	Noun + Verb → Noun	A noun that is recognized as a role by GATE and is preceded by a verb is annotated and then formed to role.	COBIT PO4.8 Obtain direction from senior management	Role: senior management

Table 30: Rules to form ISM roles

6.4.3.5 Final-Normalizer

Based on the annotations received from the instantiated GATE application, the `Verb Lemmatizer` within the `Final-Normalizer` finalizes the normalization started by the `Pre-Normalizer`. It calls the Dragon ToolKit lemmatizer to identify the lemmas for verbs in the passive form and gerunds. Lemmatization is the process of reducing a word to its canonical form. The canonical form is also a valid word that is called lemma [Nugues 2006]. For example, the canonical form of the following words "going", "lying" and "gone" is "go", "lie" and "go" respectively. There is a GATE plug-in that supports the lemmatization task, but it can be used only under the UNIX system. Therefore, we used the Dragon Toolkit within the Java application to perform this final normalization.

6.4.3.5.1 Normalization Rules

Based on the writing style guidelines presented in the section above, the `Verb Lemmatizer` performs the second and final step in the normalization for the rule 3 and 4 (Table 31 - rule 3b and 4b).

Verb Tenses				
Id	**Normalization Rule**	**Description**	**Example - Original**	**Transformation**
3b	main verb in past participle → main verb in active	The verb in the passive form is normalized in its active form.	CMMI OT SG2: Training for individuals (…) is provided.	Provide
4b	Verb in present continousin active (auxiliary verb + -ing form of the main verb) → -ing form of the main verb	The verb in the present continuous form is normalized in its active form.	COBIT PO4.11 (..) personnel are performing only authorised duties.	Perform

Table 31: Normalization rules – Second step

6.4.3.6 Example

Table 32 illustrates the application of the forming rules on the CMMI-DEV practice from our example scenario. No normalization rules need to be applied.

ISM practice	Rules		Extracted ISM practice elements
	Normalization	Forming	
CMMI-DEV PPQA SP2.1 Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers	-	Activity 1.1	communicate quality issues
	-	Output 4.2	communicated quality issues
	-	Input 3.2	quality issues
	-	Activity 1.1	ensure the resolution of noncompliance issues
	-	Output 4.1	noncompliance issues
	-	Role 5.1	staff
	-	Role 5.1	managers

Table 32: Example – Extraction of ISM practice elements based on extraction rules

6.5 Summary

The MOSAIC Toolbox is a web application that integrates a series of tools to support the modeling and analysis activities.

It offers different possibilities to model the ISM, ICM and SFM elements: the Modeler can be guided stepwise and model the data directly in the database or can model the data in XML format and then import it in the database. Furthermore, it offers a tool that semi-automatically extracts the ISM practice elements based on the ISM practice.

Based on these models, the Analyzer can automatically select ISM practices based on SFM situationalFactors, identify the similarity degree, coverage degree, output states of ISM practices and dependencies between them.

7 Applications

In this chapter, we give an overview of possible applications of MOSAIC for organizations that are interested in the adoption of multiple PRs and assessment based on PRs. We call these possible applications of MOSAIC **usage activities**.

As organizations use different types of PRs, we remind about their definition.


Reminder

A **reference PR** is used as a guideline for the software process improvement of organizations. For example, CMMI-DEV, COBIT or ISO/IEC 12207 are reference PRs.

A **process PR** refers to a process model and is more concrete than a reference PR because it defines not only ISM practices, but also give additional information about how to adopt these. It can be used as a guideline, but it can also be directly applied to describe the software processes of an organization. For example, V-Model XT is such a PR.

An **internal process PR** is a special process PR and refers to the internal software processes of an organization. This PR defines activities to be used as guidelines for the improvement of software processes in an organization, e.g. in software projects. Hence, it defines the practices of this organization and thus, it is a PR.

To offer a relevant list of usage activities, we interviewed various experts that work in various organizations and have experienced different usages of PRs. We performed two types of individual interviews with these experts:

- We performed individual interviews with eight experts about possible applications of MOSAIC. After a short description of MOSAIC, we asked them about possible usage activities for the MOSAIC models and the analysis activities: selection of ISM practices based on the project context, identification of similar ISM practices and identification of dependencies between ISM practices.
- We performed interviews with seven software process assessors. We asked them about the assessment process in general and not about possible MOSAIC applications. The experts described their method to perform assessments by answering questions, such as: how do they prepare for an assessment, which strategy are they follow to perform such an assessment, do they consider the software project context or how do they prioritize the improvement practices to be adopted in software projects. Based on these questions, we analyzed the assessment process and verified if MOSAIC can support this process in particular.

Table 33 illustrates the profiles of the interviewed experts. Firstly, we involved industrial software consultants specialized in supporting organizations in the usage of PRs. They play various organizational roles in the software process improvement initiatives. Secondly, we involved various roles from two organizations, organization A and B (described in section 8.2). We interviewed a software process improvement sponsor, process improvement lead, process manager and process engineer from the organization A. These organizational role names correspond to the roles defined and

used in the software process improvement according to CMMI-DEV [Kulpa 2003]. At the time of the interviews, we could not involve any software process assessors from this organization as this did not perform such assessments. However, we involved software process assessors from organization B. They were responsible to perform assessments and to continuously evaluate the process compliance of software projects according to the internal process PRs and CMMI-DEV.

ID	Experts	Organizational role	Interview scope	PRs knowledge					Process experience (years)
				CMMI-DEV	SPICE	CMMI-SVC	ITIL	COBIT	
1	Consultants	Various organizational roles	MosAIC Application	✓	✓	✓	✓	✓	> 10
2			MosAIC Application	✓	✓	✓	-	-	> 10
3			MosAIC Application	✓		✓	-	-	>10
4			Assessment	✓	✓	-	-	-	> 10
5			MosAIC Application	✓	✓	✓	-	-	>10
6			Assessment	✓	✓	-	-	-	>10
7			Assessment	✓	-	✓	-	-	>10
8	Industrial partners Organization A	Software process engineer	MosAIC Application	✓	-	-	-	-	3
9		Software process manager	MosAIC Application	✓	-	-	✓	-	5
10		Software process improvement lead	MosAIC Application	✓	-	-	-	-	8
11		Process improvement sponsor	MosAIC Application	✓	-	-	✓	-	3
12	Industrial partners Organization B	Software process engineer	Assessment	✓	✓	✓	✓	-	9
13		Software process assessors	Assessment	✓	-	-	-	-	3
14		Software process assessors	Assessment	✓	-	-	-	-	2
15		Software process assessors	Assessment	✓	-	-	-	-	2

Table 33: MOSAIC applications - Experts profile

In the following, we give an overview of the collected usage activities and describe them in details.

7.1 Overview

Based on the interview results, we identified several usage activities where MOSAIC can be applied. We observed that these are related to the two main challenges mentioned in the introduction of this work (section 1.3). Consequently, the experts implicitly acknowledged the fact that these two challenges exist when dealing with PRs:

- Selection of the best PRs based on the internal needs and problems of organizations
- Simultaneous usage of multiple PRs to adopt and assess the adoption by the identification of similarities and dependencies of the selected PRs

As the selection and usage of a PR do not necessarily mean the selection and usage of the entire PR, we remind about it.


Reminder

The term **PR** does not necessarily refer to all elements of a certain PR, but can also refer to a subset of elements of this certain PR that are selected for the software process improvement in an organization.

In the following, we give an overview of the usage activities. A more detailed description that explain their purpose is given in the next sections. If necessary, we give examples for a better understanding of these activities.

For the selection of the best suited PRs, we identified the following usage activities that can be supported by MOSAIC:

- *Identify the process profile of the organization*: Based on the organizational context (inclusively the needs or problems of software projects), create a process profile of ISM practices or processes to be used as reference for the improvement of the internal process PRs.
- *Identify the value of PRs*: Based on the organizational context (inclusively needs or problems of software projects), identify the value of a reference PR for an organization:
    - Compare the values of reference PRs of interest to decide which reference PR(s) should be adopted.
    - Based on the value of a reference PR for an organization and the effort needed for the adoption, compute its ROI (return on investment) for this organization to decide if this reference PR should be adopted.
    - Identify the additional value of a reference PR (Fig. 47 - marked in white) to decide if it should be adopted.
    - Identify the value that can be lost (Fig. 47 - marked in white) when the internal process PRs remain compliant to a reference PR_1, but not to PR_2 anymore.

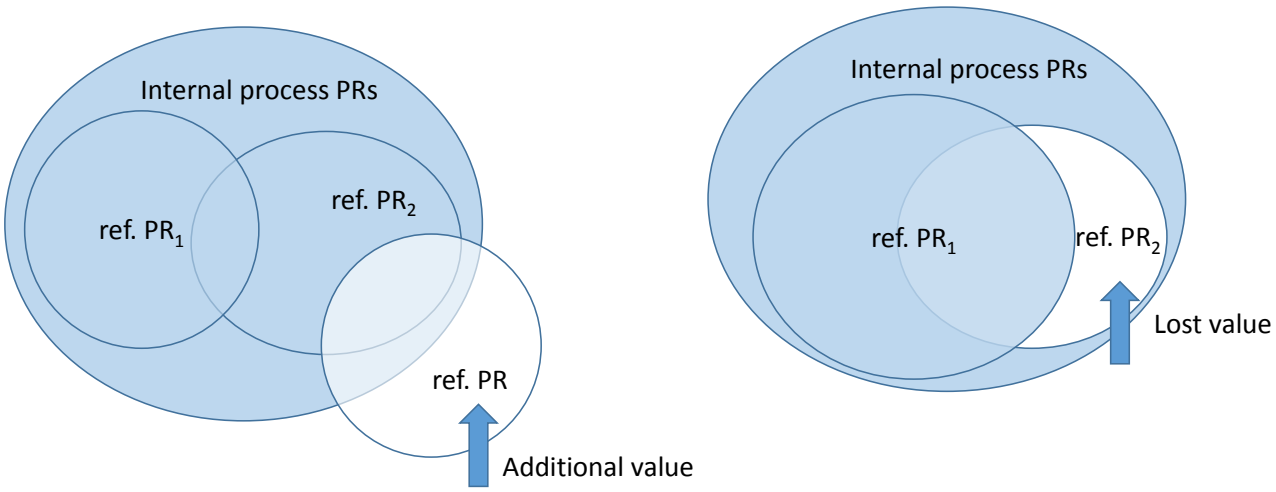


Fig. 47: Additional and lost value in the adoption of reference PRs

- *Establish a new PR*: Based on a PR for an IT domain, derive a new PR for another IT domain that is needed in an organization.
- *Create a tailoring instrument for software projects*: Based on a mapping between the software project context and the practices of the internal process PRs, create a tailoring instrument to recommend best suited ISM practices that need to be adopted to address critical situations in this software project.

Once the best suited PRs are selected for the adoption, there are different activities that describe the simultaneous usage of these PRs (inclusively internal process PRs) and can be supported by MOSAIC:

- *Create a repository of multiple PRs*: Create a repository that covers multiple PRs and supports the organization in an efficient adoption and assessment.
- *Avoid redundancies:*
    - Avoid redundancies in the various ISM categories of the internal process PRs.
    - Identify the redundancies of two or more reference PRs to be used for the improvement of the internal process PRs.
    - Identify the overlapping between a process PR, an internal process PR and reference PRs when a process PR is considered to substitute the internal process PR.
- *Avoid inconsistencies*:
    - Assure a consistent definition of highly dependent ISM practices in the internal process PRs.
    - Verify whether the dependencies between ISM practices are consistent.
- *Provide helpful information for adoption*:
    - Use the additional information from similar reference PRs to improve the internal process PRs.
    - Identify which ISM practices from different PRs are abstract and which are more concrete to better understand and adopt ISM practices that need to be considered.
    - Identify the time order of the ISM practices in the internal process PRs to give guidelines for their adoption in the projects or other parts of an organization.
    - Achieve a common terminology in the internal process PRs.
- *Maintain the internal process PRs*: Identify the global impact of the local changes performed in some ISM practices to avoid negative effects. If the negative global effects are higher than the local optimization, then the local change should not be performed.
- *Achieve compliance to reference PRs*: Improve the internal process PRs according to a reference PR based on the gaps between them.
- *Perform efficient assessments*:
    - Perform assessments by simultaneously considering multiple PRs to avoid redundancies in the evaluations.
    - Perform assessments by considering the dependencies between the ISM practices to avoid unnecessary evaluations.

Another usage activity that is mentioned by the experts concerns the redundancies and conflicts of software requirements in one or more requirements specifications. These redundancies and conflicts have to be avoided. This activity can be supported by MOSAIC, but it is not related to the usage of PRs. Therefore, it is not listed above.

The afore-mentioned activities partially overlap or support each other. For example, the activity "achieve compliance to reference PRs" supports the activity "identify the additional value of reference PR". The activity "create a repository of multiple PRs" partially overlaps with the activity "define internal process PRs without redundancies". We listed each activity to emphasize their importance as individual activities that can be considered by organizations that deal with PRs.

In the following, we describe and motivate each usage activity in detail. We continue with guidelines on how MOSAIC can be applied to support these activities. Limitations for the MOSAIC application are also mentioned. We differentiate between limitations for the approach and limitations for the tool. We defined various algorithms that are implemented in the MOSAIC Toolbox. These algorithms are only examples of implementing the MOSAIC analysis activities (chapter 5). However, there are various algorithms for analyzing PRs that can be developed by applying the MOSAIC metamodels and metrics.

7.2 Process Profile of an Organization

A process profile defines the ISM practices or processes from one or multiple PRs that are best suited for an organization and can be used as reference for the adoption and assessment.

The continuous representation of the CMMI constellation mentions such process profiles that refer to different process areas. Within this representation, process areas have to be selected according to the business objectives of the organization and adopted until a certain capability (0-5) is achieved. For example, the CMMI-DEV "Supplier Management" process area can be selected to be a part of such a process profile of an organization, when the business of the organization highly depends on the systems or software developed by third parties.

Consequently, the benefit of such a process profile is that it considers the ISM practices or processes that are relevant for an organization based on its current situation.

7.2.1 MOSAIC Application and Limitations

To create such a process profile for an organization, the ISM practices and its corresponding ISM processes have to be selected based on the context of the organization. MOSAIC can be used to define such a process profile based on the software project context. The selection of ISM practices based on SFM situationalFactors identifies ISM practices from multiple PRs that are best suited for an organization.

For this purpose, the context of software projects has to be analyzed and characterized by SFM situationalFactors. The most frequent SFM situationalFactors that are critical or apply to these projects can be then used to select the best suited ISM practices and define the process profile. In collaboration with one organization, we analyzed the context of three software projects and selected ISM practices that are important for this organization (section 8.4.3).

A limitation of MOSAIC is that it only considers the selection of PRs for the software development. The reason is that the SFM situationalFactors are derived from the situational factors framework [Clarke and O'Connor 2012] that describe the software development settings. However, we argue that these SFM situationalFactors are also valid for other software areas. For example, the SFM situationalFactors "personnel cohesiveness" or "customer satisfaction" from the "personnel" category

could be considered to characterize each type of project or work and not only software projects. However, a systematic analysis of the SFM situationalFactors validity has to be performed in the future work.

Another limitation of MOSAIC is that we consider only the bottom-up approach to describe the context of an organization. The bottom-up approach considers the software project context to describe the needs and constraints of an organization. However, the top-down approach based on the organization's goals might also be relevant for the definition of the process profile an organization. Therefore, a selection of ISM practices or processes based on the organization's goals is needed. Based on our experiences with organization's goals, we assume that the SFM situationalFactors can also be used to describe these. However, this fact has to be closely analyzed in the future work. Furthermore, there exist various approaches that address the selection of ISM processes based on goals (section 1.3.1). These approaches can be analyzed and integrated into MOSAIC to support a better selection and definition of the process profile of an organization.

To summarize, MOSAIC selection of ISM practices based on SFM situationalFactors can be applied to identify the process profile for an organization. A limitation is the validity of the SFM situationalFactors for other software areas. Moreover, the lack of consideration of the organization's goals is also a limitation of MOSAIC. The relation between organization's goals and SFM situationalFactors need to be analyzed to verify if MOSAIC is suitable for this goal based selection.

7.3 Value of a Reference PR

The value of a reference PR for an organization refers to the amount of the ISM practices of this reference PR that are best suited for this organization. There are several situations where the value of a reference PR should be considered.

If an organization needs to decide which reference PRs have to be used for its process improvement, the value of these reference PRs can support the decision process. For example, the interviewed experts mentioned that many organizations have difficulties to decide whether they should use CMMI-DEV, SPICE or ISO 12207 to improve their software processes.

Furthermore, this decision can also be supported by an analysis of the value of a reference PR and the effort needed to adopt it, i.e. by an analysis of the ROI of this reference PR. If the value of a reference PR is higher than the value of other reference PRs, but the ROI is smaller, then the organization should be aware of this fact and consider it in the decision process.

The additional or lost value of reference PRs can also support organizations in the decision process. Organizations aim to be compliant to more than one reference PR for a certain software area to increase their competitive strength on the market or to fulfill the requirements of a customer:

- *Additional value*. If an organization is compliant to a reference PR and needs to decide to be compliant to another reference PR for the same software area, the additional value for this gap has to be estimated. This helps the organization to decide if this additional adoption is convenient or not. For example, an organization adopts already CMMI-DEV, but the customer requests the SPICE compliance. The gap or difference between CMMI-DEV and SPICE has to be identified to decide if the adoption of this gap is valuable or not.
- *Lost value*. When an organization needs to decide, if it is not convenient to remain compliant to multiple reference PRs for a certain software area, the lost value have to be analyzed. If the effort needed to maintain this process compliance is higher than the value that can be lost, then the organization can decide to not remain compliant to some PRs.

7.3.1 MOSAIC Application and Limitations

To determine the value of a reference PR for an organization, the best suited ISM practices for this organization need to be identified. The selection of ISM practices based on SFM situationalFactors in MOSAIC helps to identify the ISM practices that can be considered for this value. Based on their support degree, on the number of selected ISM practices and the total number of ISM practices of a reference PR, the value of this reference PR can be calculated. This is not currently implemented and thus, it is a limitation that can be addressed in the future work.

To compute the additional or lost value, the gap between the reference PRs have to be considered. The identification of the gap is supported by MOSAIC. However, the MOSAIC Toolbox cannot automatically identify this gap. An Analyzer has to manually analyze the computed coverage or similarity degree to identify which ISM practices are in the gap. Based on the support degree and the number of these ISM practices in the gap, the additional or lost value can be calculated. Analogously to the value of a reference PR, this is a limitation of MOSAIC and can be addressed in the future work.

To summarize, MOSAIC with its selection of ISM practices based on SFM situationalFactors and identification of similar ISM practices can be used but it has to be extended to compute the value of a reference PR. Although we give some guidelines how this value can be calculated, this computation has to be systematically developed and evaluated. Furthermore, the identification of the gap between reference PRs is supported, but can only be semi-automatically identified with the support of the MOSAIC Toolbox.

7.4 New PRs

New PRs can be established based on the knowledge of existing PRs. The reuse of existing PRs is actually a fact. For example, an analysis of 52 PRs reveal that the most of them are based on CMM, SPICE or CMMI-DEV [von Wangenheim et al. 2010].

This is beneficial as the existing PRs reflect the long years' experience of various organizations. There are different domains and not for every domain there is a corresponding PR to be used. For example, the software development in the health care industry domain becomes more and more important and thus, best practices are needed to guide this software development for medical devices. Consequently, new PRs for the health care industry are defined, e.g. MediSPICE is defined based on ISO/IEC 12207, ISO/IEC 15504 and IEC 62304 [O' Malley 2013].

The definition of new PRs based on existing PRs can be time-consuming. For this definition, various similar PRs and the target domain have to be intensively analyzed. Therefore, techniques are needed to support this process.

7.4.1 MOSAIC Application and Limitations

An approach, called the translation of a PR, is proposed to create a new PR by transforming an existing PR into a new one [Fricker et al. 2013]. Here, the concepts of the existing PR are analyzed and translated into similar concepts that are specific to the new domain. For example, the concept "customer" in SPM Framework (Software Product Management Framework) is translated to "patient" in the new PR for the health care industry [Pettersson et al. 2008].

MOSAIC with its ICM can be useful to support this translation. ICM similar concepts of an existing PR and of the new PR can be modeled in the ICM. Based on these similarity relations between

the ICM concepts, ISM practices can be automatically specified by replacing the ICM concepts of the existing ISM practices with ICM similar concepts from the target domain. This is currently not implemented in MOSAIC and thus, it is a limitation.

To summarize, MOSAIC with its models can support the definition of new PRs based on existing PRs. We did not implement such an automated transformation, but MOSAIC can be used to support it.

7.5 Tailoring Instrument for Software Projects

A tailoring instrument for software projects defines which ISM practices are best suited for these projects and thus, supports the project members to select ISM practices of the internal process PRs.

For example, if the requirements volatility is high in software projects, then practices, such as the SPICE SUP.10 BP7 "Analyze and prioritize the change requests" or CMMI-DEV REQM SP1.4 "Maintain bidirectional traceability among the requirements and work products" have to be addressed as possible mitigation actions to manage the multiple change requests that appear during the software project. If the change requests are not analysed and prioritized, then the effect that this change has on the development is unknown and can jeopardize the project goals. Furthermore, if the traceability among requirements and work products is not defined, then the changes cannot be effectively and efficiently managed. The design components, the implementation or the test cases that are affected by these changes need to be identified each time such a change is requested. For a high requirements volatility, this requires a big effort without such a traceability between the requirements and the work products.

Therefore, based on the context of software projects, PRs are adopted to support these projects to address different situations. This leads to an effective adoption of the internal process PRs as only ISM practices that are beneficial are adopted. Moreover, the software projects recognize this benefit and the process compliance improves.

7.5.1 MOSAIC Application and Limitations

A possible implementation of a tailoring instrument for software projects is to define a mapping between factors that characterize the project context and the internal process PRs' practices. Within MOSAIC, the selection of ISM practices based on SFM situationalFactors can be applied to create such a mapping. For each SFM situationalFactor, the ISM practices with their support degree is identified. In collaboration with our industry partner, we created such a tailoring instrument and applied in the tailoring process of software projects (section 8.4.3). There is no limitation for MOSAIC to support the creation of such a tailoring instrument.

7.6 Repository of Multiple PRs

A repository of multiple PRs contains all the ISM practices of the PRs of interest. For each ISM practice, the traces to PRs, the similarities, differences and dependencies between the ISM practices can be identified in such a repository. Furthermore, the process compliance according to these PRs can also be documented here.

One of the PRs contained in such a repository can be the internal process PR. This increases the understanding and the acceptance of the multiple PRs to be adopted in the organization. The similarities between the internal process PRs and other ISM practices lead to a better understanding of these PRs.

There are several advantages of creating such a repository of multiple PRs.

One advantage is that such a repository of multiple PRs helps an organization to avoid redundancies in the adoption and assessment. It gives an overview of the similarities and the differences between the multiple PRs. This is time-efficient as ISM practices from different PRs need to be considered and assessed only once. For example, the results of an assessment according to a PR can be partially used to verify the compliance according to another PR when these PRs have redundancies.

A repository of multiple PRs can also support an organization to manage the communication between the different departments responsible for the different PRs. The information flow between the departments is known and this supports an efficient communication.

Another advantage is that such a repository supports a better understanding of the process needs and wishes of an organization by giving an overview of the PRs that are currently considered in the organization. A better understanding is also supported by a common structure and terminology of the multiple PRs.

Finally, this repository can help an organization that offers consultancy services about the usage of multiple PRs to deliver their customers information about the similarities, differences and dependencies of PRs of interest.

7.6.1 Examples of Reference PRs for a Repository

We give some examples of commonly used reference PRs that can be considered for integration into a repository of multiple PRs. There are different publications that mention which are most commonly used PRs [ISACA 2011c; Heston and Phifer 2011]. We present the results of a survey documented in the Global Status Report on the Governance of Enterprise IT (GEIT). This survey involves 834 business executives of small and large IT organizations of different types from 21 countries. According to this survey, ITIL or ISO 20000 are used by 28% of the organizations and thus, are most widely used PRs (Fig. 48).

A repository that contains some of these commonly used reference PRs can be interesting for a consultant organization as most of its customers use these reference PRs.

Furthermore, these can be relevant for an organization as these commonly used reference PRs cover the process landscape of an IT organization and thus, can be used as a reference for its whole process improvement. For example, CMMI-DEV can be used for the software development, ITIL or ISO 20000 for the service operation, COBIT for the IT governance and TOGAF for the enterprise

architecture management. However, we recommend organizations to individually decide which PRs or parts of the PRs are best suited for them based on their needs.

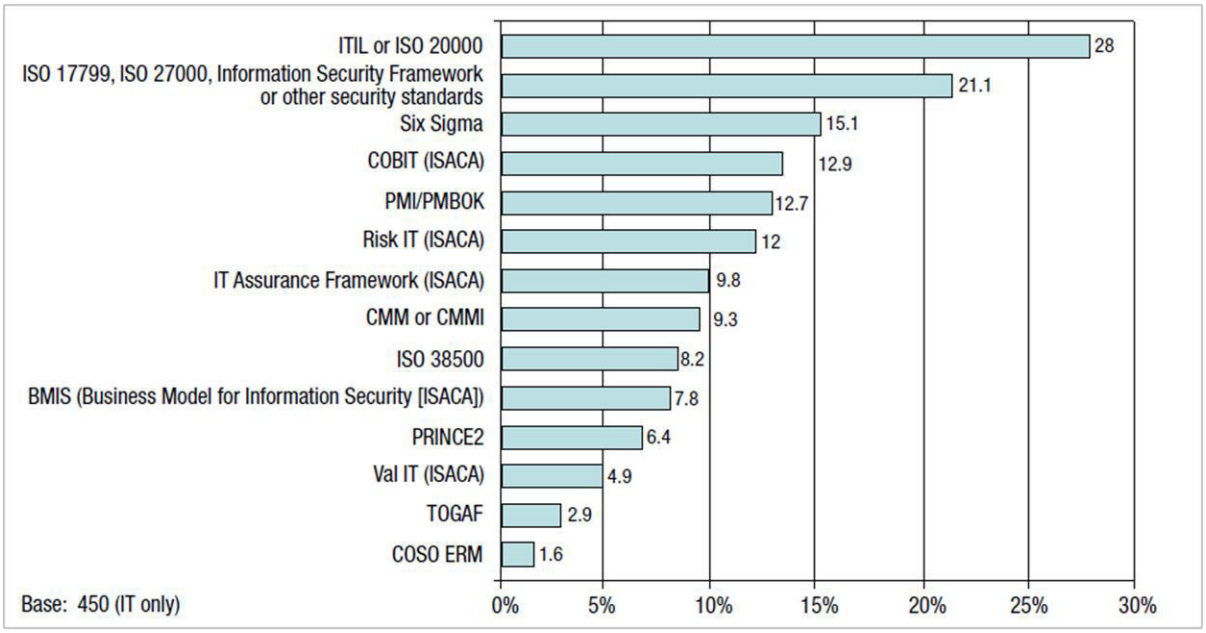


Fig. 48: GEIT – Percentage of usage of various PRs

7.6.2 Alternatives for the Construction of a Repository of Multiple PRs

There are various alternatives to implement such a repository of multiple PRs. Based on our experiences, we roughly describe some alternatives how to create such repositories. For these alternatives, we describe in the next section how MOSAIC can support them.

The multiple PRs can be integrated into the repository as they are originally defined. However, we recommend integrating the PRs according to a common structure and terminology. This leads to a better understanding of the PRs. Furthermore, the similarities and differences between the PRs are easy to identify due to this common structure and terminology. Otherwise, a mapping between the PRs might be quite complex [Rout and Tuffley 2007].

Another alternative is to create a repository that contains unique ISM practices that are derived from the ISM practices of the multiple PRs. For each unique ISM practice, the traces to the PRs and dependencies to other unique ISM practices are given. For example, the internal process PRs' practices can be considered as unique ISM practices. This leads to a better understanding and acceptance of the PRs within an organization that uses the repository for their process improvement. We created a repository of multiple PRs within the field study performed with our industry partner that contains such unique ISM practices and the references to an internal process PR and CMMI-DEV (page 189).

An enhancement of the last alternative is to differentiate between abstract and special unique ISM practices in the repository of multiple PRs. For example, the repository of multiple PRs can contain the special ISM practices "Establish the overall project plan" derived from CMMI-DEV PP SP2.7 practice and "Establish the work plan" derived from CMMI-SVC WP SP2.7 practice. For these two special ISM practices, such a repository also contains their abstract ISM practice "Establish the plan".

In the future work, a systematic analysis has to be performed to analyze other alternatives to create such a repository. Another future work could be to enhance the repository of multiple PRs with guidelines for its use. For example, we make an analogy between PRs' categories or processes and software components. Then, we can state that the architecture of a repository of multiple PRs is described by these components and dependencies between them. Therefore, an analogy between such a repository and a software system can be useful. Inspired by architecture design principles, such a repository of multiple PRs can be created. For example, the dependencies between process components can be defined according to rules of cohesion and coupling for software components. Therefore, an efficient information flow between the departments can be better supported. The future work could consist of an intensive analysis of this analogy and definition of guidelines from software architecture design theory to be used to create and maintain this repository of multiple PRs.

7.6.3 MOSAIC Application and Limitations

There are various alternatives to create a repository of multiple PRs and thus, MOSAIC can differently support the creation of such repositories.

One alternative is a repository that contains all PRs practices that are organized according to a common structure and terminology. To create a repository with a common structure and terminology, MOSAIC can be applied to achieve this. The IS Meta-Model can be used to normalize the structure and the IC Meta-Model to normalize the terminology.

A repository with abstract and special ISM practices is also an alternative. With MOSAIC, we can support the definition of such ISM practices based on the identification of ICM abstract concepts. These ICM abstract concepts are identified by the MOSAIC Toolbox when the similarity degree between two or more ISM practices is computed. However, the extraction of abstract ISM practices cannot be automatically performed and thus, the MOSAIC Toolbox have to be extended.

For both alternatives and other alternatives, such as a repository that contains the PRs as they are originally defined or a repository that contains unique ISM practices, the similarities, differences and dependencies between the ISM practices have to be identified.

With MOSAIC, we can identify the similarities and differences between PRs by analyzing the similarity degree, coverage degree or the output states of their ISM practices. There are limitations for the MOSAIC Toolbox as there are various possibilities how to present the similarities and differences between ISM practices. In some cases, an Analyzer has to manually analyze the MOSAIC Toolbox results to get the information in the needed form. Therefore, based on the needs of an organization, the MOSAIC Toolbox has to be extended. Here are some examples of analysis activities that can be partially or completely supported by the MOSAIC Toolbox:

- *Identify all the similar ISM practices of an ISM practice*. For a single ISM practice all its "Equal", "High", "Medium" or "Low" ISM practices from other PRs can be automatically identified by the MOSAIC Toolbox.
- *Identify the mapping between two PRs*. With the support of the MOSAIC Toolbox, we can semi-automatically deliver this mapping. For each ISM practice of the PR used as reference, its "Equal" or "High" ISM practices are automatically identified. These have to be analyzed and added to the mapping.
- *Identify the differences or the gap between two PRs*. With the support of the MOSAIC Toolbox, we can semi-automatically identify the gap. Based on the similarity degrees "High", "Medium" or "Low" between two ISM practices, a manual analysis is needed to identify the differences between these ISM practices. A manual analysis of the computed coverage degree can also be

performed to identify the gap. If there are no ISM practices from one PR that cover one ISM practice from another PR, then this last ISM practice is in the gap between the two PRs.

A repository with multiple PRs contains the dependencies between its ISM practices. With MOSAIC, we can identify the dependencies for each ISM practice. This is also implemented in the MOSAIC Toolbox. There is no limitation to be mentioned.

The identification of similarities and dependencies based on MOSAIC metrics is time-consuming due to the detailed results. The computation of the results based on these metrics is performed at the fine-grained level of ICM concepts. Therefore, the results are differentiable and detailed. For example, the similarity degree between the ISM outputs of the considered ISM practices can be identified. Furthermore, this fine-grained structure allows an automation. However, this requires a high computation time and depending on the number of selected practices, an Analyzer have to wait for the computed results.

To summarize, MOSAIC can be used to create such a repository of multiple PRs. The following parts of MOSAIC are relevant for this purpose:

- IS and IC-Meta Models to achieve a common structure and terminology of the multiple PRs
- Identification of similar ISM practices to identify the similarities and differences of the ISM practices
- Identification of dependencies between the ISM practices to define the information flow between the different parts in the repository of multiple PRs

However, MOSAIC could be improved or extended to satisfy the special needs of an organization concerning the presentation of the similarities and differences. Furthermore, an automated extraction of abstract and concrete ISM practices can also be an extension of the MOSAIC Toolbox.

7.7 Avoid Redundancies

Redundancies of multiple PRs refer to their similar ISM practices.

Two or more reference PRs to be adopted can have redundancies and these need to be avoided. For example, CMMI-DEV and SPICE have a high number of redundancies because both PRs define guidelines for the management and development of software products. The redundancies of multiple PRs have to be avoided for an efficient definition and improvement of the internal process PRs.

Another case is if the organization decides to implement a process PR as the internal process PR for a certain software area. For example, one of our industrial partners decided to implement the RUP (Rational Unified Process) as their internal process PR. Moreover, another goal was to be compliant to CMMI-DEV. In this case, the organization needed to know the intersection between their existing internal process PR, between RUP and CMMI-DEV. Therefore, the experience reflected by the internal process PR did not get lost and the redundancies between all these three PRs were avoided.

There can also exist redundancies in the different ISM categories of the internal process PRs. For example, we discovered such a redundancy in the internal process PR of our industry partner. In all its ISM categories “project management”, “test”, “software development” and “quality assurance”, the ISM practice “Perform a lessons learned workshop” with all its additional information (e.g. guidelines how to perform such workshops or templates) was defined. This is a redundancy. A possible solution to avoid this could be to define this ISM practice with all its additional information only once and reference it in the different ISM categories.

7.7.1 MOSAIC Application and Limitations

To identify the redundancies between PRs, we can use MOSAIC to identify similar ISM practices by analyzing the similarity degree, coverage degree or the output states of their ISM practices.

Two or more ISM processes can be selected in the MOSAIC Toolbox and the similarity degrees of all possible combinations of their ISM practices from different PRs are automatically computed. If the similarity degree is “Equal”, “High” or “Medium” then there is a redundancy between the considered PRs that need to be considered and analyzed. However, this computation is time-consuming due to the consideration of all possible combinations of ISM practices.

Furthermore, the *highest coverage* and *best coverage* can be used to avoid redundancies and perform a time-efficient adoption of more ISM practices.


Reminder

The **highest coverage** refers to a practice that has the maximum coverage degree in a considered set of practices.

The **best coverage** refers to the minimum subset of practices with a coverage degree of 1 in a considered set of practices.

If an organization needs to adopt more ISM practices, it can choose the ISM practice with the highest coverage degree to cover as much as possible from the entire set of practices. Therefore, redundancies are avoided. Redundancies are also avoided in the adoption of the subset of ISM practices with the best coverage degree. An organization can only concentrate on these subset of ISM practices to cover the entire set of ISM practices. Consequently, it is time-efficient as the content of all ISM practices are addressed, but only a minimum subset is considered.

A limitation for the MOSAIC Toolbox is the identification of the redundancies in a single PR. In the current version of the MOSAIC Toolbox, we can identify the redundancies between two different PRs. This can be easily changed in the future work.

7.8 Avoid Inconsistencies

Inconsistencies in the internal process PRs refer to an inconsistent description of its elements or inconsistent relations between these elements. The inconsistencies in the internal process PRs have to be avoided for an efficient and effective adoption of the internal process PRs in the organization. We describe some examples of inconsistencies that need to be considered.

The organization has to verify whether the dependencies between ISM practices are consistent or not. Fig. 49 illustrates an inconsistent definition of dependencies between ISM practices, namely the ISM practices form a cycle and this is illegal.

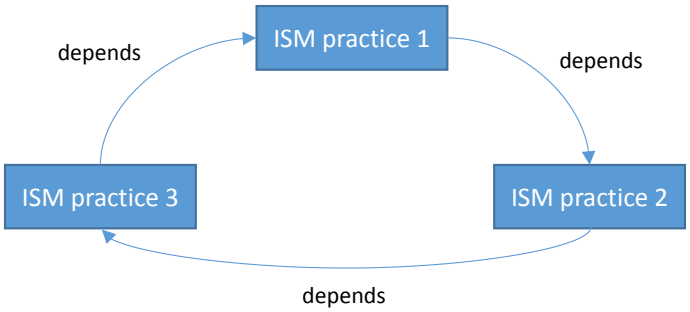


Fig. 49: Example of an inconsistency in PRs

Furthermore, the organization has to assure a consistent definition of highly dependent ISM practices in the internal process PRs. This needs to be considered when improvements are only performed on some selected ISM practices of the internal process PRs. Not only these ISM practices, but also their high dependent ISM practices have to be considered to avoid inconsistent descriptions. For example, improvements of a template that describes the output of an ISM practice can lead to the deletion of some content that is relevant for another ISM practice.

7.8.1 MOSAIC Application and Limitations

To assure a consistent definition of internal process PRs, the dependencies between its ISM practices can be analyzed. The identification of dependencies in MOSAIC can be used for this purpose. This is implemented by the MOSAIC Toolbox. In the future work, an automated analysis of all dependencies and identification of such faults can be implemented.

We have mentioned so far only inconsistencies related to dependencies between ISM practices. A systematic analysis can be performed in the future work to identify other types of inconsistencies and tool support to automatically locate them.

To summarize, MOSAIC with the automated identification of dependencies between the ISM practices supports the identification of inconsistencies. With the MOSAIC Toolbox, we cannot perform an automated analysis of the inconsistencies and thus, this is a limitation.

7.9 Provide Helpful Information for Adoption

Helpful information in the internal process PRs refers to information about the adoption of its ISM practices in software projects or other parts of an organization. We name some examples of such helpful information.

Methods on how to perform the ISM activity described in the ISM practice, a template for its ISM output or the lifecycle of this ISM output can be such helpful information. The knowledge contained in the existing process or reference PRs can be used to identify this helpful information. For example, if an organization needs information about how to analyze the software project stakeholders, then the description of the CMMI-DEV PP SP2.6 “Plan the stakeholder involvement” can be useful: “A two-dimensional matrix with stakeholders along one axis and project activities along the other axis is a convenient format (…). Relevance of the stakeholder to the activity in a particular project phase and the amount of interaction expected would be shown at the intersection of the project phase activity axis and the stakeholder axis”.

The time order of ISM practices is also helpful for the adoption. For example, this helps software projects to know what to consider next in a certain phase of the project.

Furthermore, a dictionary of the terms used in the internal process PRs are also useful for the adoption. This leads to a better understanding of the ISM practices and thus, to a better adoption.

7.9.1 MOSAIC Application and Limitations

Helpful information contained in the existing process or reference PRs can be identified by analyzing the similarities between their ISM practices and the internal process PRs practices. MOSAIC supports this identification of similar ISM practices.

The computation of the similarity degree or of the output states are the most relevant analysis activities for this purpose.

The similarity degree between ISM practices of an internal process PR and a process or reference PR can be used. For example, methods or typical work products for an internal process PR practice can be found in the description of "Equal" and "High" ISM practices from process or reference PRs. Furthermore, "Medium" or "Low" ISM practices can be analyzed to identify the differences between them. These differences can reveal useful information.

The computation of the output states in the MOSAIC Toolbox delivers information about the lifecycle of the ISM output, i.e. about its creation, implementation and verification. This is a valuable information for the adoption of a required ISM output.

An important information in the internal process PRs is the time order of the ISM practices. MOSAIC supports this by the identification of dependencies between the ISM practices. The MOSAIC Toolbox implements this without any limitation to be mentioned.

MOSAIC also supports the creation of a dictionary of terms used in the internal process PRs. The ICM with its ICM concepts, similarity relations between them and the relations to the ISM outputs, inputs, roles and purposes can be used to create such a dictionary.

To summarize, MOSAIC with the identification of similar ISM practices and dependencies between them support the identification of helpful information that can be integrated in the internal process PRs. Furthermore, its ICM can support the creation of a dictionary of terms used in the internal process PRs. There is no limitation to be mentioned.

7.10 Maintenance of Internal Process PRs

The maintenance of the internal process PRs refer to the changes performed in this PR to achieve different goals. The internal process PRs have to be continuously maintained to respond to the changes in the business processes of an organization or to fulfill customer, legal or regulatory requirements.

When the internal process PRs are maintained, the organizations need to pay attention to the impact these changes have. The global impact of the local changes performed in some ISM practices has to be identified to avoid negative effects. Changes of some templates, methods or supporting tools can have a negative effect for the different departments that work in the different software areas of an organization. If the negative global effects are higher than the local optimization, then the local change should not be performed. For example, such changes can lead to the deletion or modification of some ISM categories or processes so that the internal process PRs loose the process compliance according to reference PRs. This could jeopardize the cooperation between the organization and the customer that requests this compliance. More, it can lead to a cancelation of the contract between

them. For example, the compliance according to CMMI-DEV or SPICE is important in the automotive industry and is requested by the OEMs (Original Equipment Manufacturers) for their suppliers.

7.10.1 MOSAIC Application and Limitations

The impact of the changes in the internal process PRs can be supported by the identification of the dependencies between the ISM practices. These dependencies have to be analyzed to decide if the changes in one ISM practice could lead to negative effects in its dependent ISM practices. The MOSAIC Toolbox can be used to identify "Strong" and "Medium" dependencies. The organization needs to differentiate between "Strong" and "Medium" dependencies because the impact can be different. The impact for "Strong" dependent ISM practices can be higher than for "Medium" dependencies as "Strong" dependencies share high similar ISM artifacts. There is no limitation to be mentioned.

7.11 Compliance to Reference PRs

The term "compliance" is often used in relation with reference PRs, such as CMMI-DEV, SPICE, ISO/IEC 12207. To achieve compliance to a reference PR, the organization has to adopt this PR. One prerequisite for this compliance is that an internal process PR is compliant to this reference PR, i.e. there is no gap or difference between these two PRs and that the organization is working according to its internal process PRs. The organizations aim to be compliant to different reference PRs to fulfill customer, legal or regulatory requirements or to improve their competitive strength on the market.

For example, Fig. 50 indicates that a high number of organizations aim to be compliant to CMMI. The figure visualizes how many organizations performed a SCAMPI (Standard CMMI Appraisal Method for Process Improvement) class A appraisal between 2007 and august 2013 [Keller and Mack 2013]. For example, 24.3% and 63.6% from 6.010 organizations performed such an assessment to achieve a CMMI level 2 and level 3 respectively.

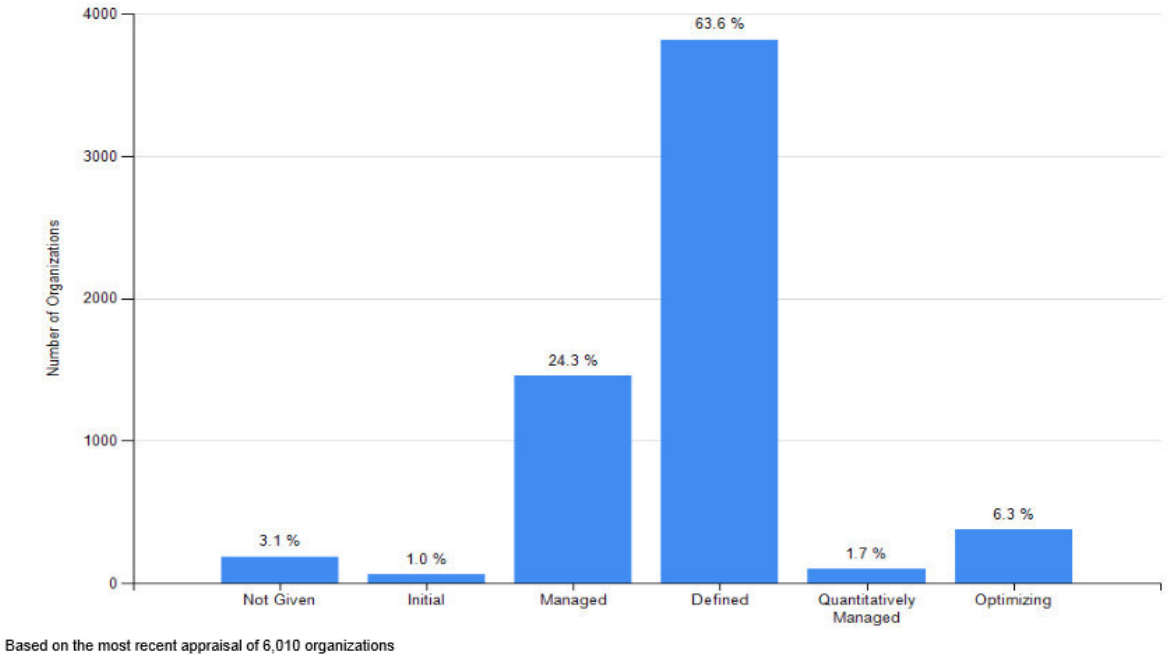


Fig. 50: SCAMPI class A appraisal results [Keller and Mack 2013]

7.11.1 MOSAIC Application and Limitations

The gaps or differences between the internal process PRs and a reference PR can be determined by the identification of low similarities between these two PRs.

With MOSAIC, we can identify these low similarities between two PRs by analyzing the similarity degree of the considered ISM practices. For example, if the similarity degree between an ISM practice of the reference PR and all ISM practices of the internal process PRs is “Non-Equal”, then the first ISM practice is in the gap and the internal process PRs are not compliant. As already mentioned, the MOSAIC Toolbox cannot automatically identify the ISM practices that are in the gap between PRs. This is a limitation and can be addressed in the future work.

Furthermore, we can identify if the internal process PRs are compliant to reference PRs by calculating the coverage degree. If the coverage degree of the ISM practices of the internal process PRs and the ISM practices of the reference PRs is 1, then the internal process PRs cover the reference PR and thus, it is compliant. However, we need to mention that this might be time-consuming if a high number of ISM practices are considered. This is because the coverage metrics are computed at the fine-grained level of ICM concepts.

To summarize, MOSAIC with the identification of similar ISM practices can be used to support organizations to achieve a compliance to reference PRs. The gap between PRs cannot be automatically identifies, but we can verify if PRs cover each other.

7.12 Efficient Assessments

During an assessment the process compliance according to PRs is evaluated, i.e. the adoption of all their ISM practices is evaluated.

To increase the time efficiency, the number of ISM practices to be considered for the evaluation has to decrease. First, if an organization is interested to evaluate the process compliance according to multiple PRs, then the assessment can be efficiently performed if these PRs are simultaneously considered. If the redundancies between the two PRs are identified, the adoption of these similar ISM practices needs to be evaluated only once. Therefore, the number of ISM practices to be considered decreases. Second, the dependencies between the ISM practices can be considered. If an ISM practice is not properly adopted then its dependent ISM practices are also not properly adopted. Therefore, the adoption of these dependent ISM practices does not have to be evaluated anymore and thus, the number of ISM practices to be considered decreases.

7.12.1 MOSAIC Application and Limitations

We can use MOSAIC to identify similar ISM practices to avoid redundancies in the assessments. For this purpose, the similarity degree, the coverage degree or the output states of ISM practices have to be analyzed. For example, if the similarity degree between two or more ISM practices is “Equal” or “High”, then the adoption of these ISM practices need to be evaluated only once. Furthermore, if one ISM practice has the highest coverage in a set of ISM practices, then this ISM practice should be assessed first to cover as much as possible of all other ISM practices. Analogously, for the best coverage degree of a set of ISM practices. This allows the identification of ISM practices that cover as much as possible from the considered ISM practices.

We can also use MOSAIC to identify which ISM practices can be skipped for the evaluation if an ISM practice is not properly adopted. This is possible by the identification of all dependent ISM practices for an ISM practice.

To summarize, MOSAIC supports this activity by the identification of similar ISM practices and of dependencies between the ISM practices. These activities are also implemented by the MOSAIC Toolbox. There are no limitations to be mentioned.

7.13 Non-redundant and Non-conflicting Requirements

Redundant requirements for the software or system development appear if there are different sources or channels from which to receive requirements. In general, there are various stakeholders (e.g. customers, end users, testers or suppliers) that have different requirements on the software product. An expert mentioned during an interview, that the suppliers in the automotive industry receive different requirement specifications from different OEMs. These overlap (the same law regulations, the same PRs that need to be adopted or the same technical requirements) so that their implementation is not efficient. Therefore, the redundancies in the requirements specifications need to be identified.

Furthermore, CMMI-DEV mentions that “Frequently, stakeholder needs, expectations, constraints, and interfaces are poorly identified or conflicting” (CMMI-DEV RD SP1.1). There are often various stakeholders that have requirements on a software project, but these requirements are not so clear, are overlapping or are conflicting. Consequently, it is difficult for the software project members to understand and clearly document these requirements. Therefore, the similarities between the requirements need to be identified to identify such conflicts.

7.13.1 MOSAIC Application and Limitations

To identify the redundant or conflicting requirements, the basic ideas of MOSAIC with its meta-models and practice similarity metrics can be used.

However, these cannot be directly applied. The MOSAIC meta-models and models need to be analyzed and eventually modified to reflect the elements of a software requirement. Furthermore, the current ICM reflects the terminology used in the PRs and not the business terminology from the different domains. Therefore, the ICM must be created from scratch or to be modified according to the target domain, e.g. automotive industry.

7.14 Summary

We performed various interviews with eight experts about possible MOSAIC applications for organizations that work with one or more PRs. Furthermore, we interviewed seven software process assessors about the assessment process to identify if MOSAIC can support this process as well. All these experts acknowledged the contribution of MOSAIC and mentioned different usage activities that are relevant for such organizations.

Table 34 summarizes the usage activities that can be supported by MOSAIC and that organizations can consider in their process improvement when working with PRs. This table can be used by organizations as a check list as it provides usage activities they need to pay attention, when working with PRs. This list can also support them in decisions related to their software process improvement program. The table also visualizes which MOSAIC analysis activities and MOSAIC models support their implementation.

For each activity, we calculated the percentage of how many experts mentioned it. We considered only the first type of interviews where we asked the experts about possible applications of MOSAIC. We did not count the answers of the software process assessors gathered in the interviews. The reason is that these interviews only focused on the assessment based on PRs. However, the interviewed software process assessors confirmed that the creation of a tailoring instrument and the identification of dependencies between the ISM practices is particularly valuable for such an assessment.

Furthermore, we do not only list the usage activities mentioned by the experts, but also other applications of MOSAIC that we think can be considered by organizations. The voting 0% means that no expert mentioned this usage activity. Furthermore, the MOSAIC models (meta-models included) support each usage activity as the analysis activities are based on them. However, there are usage activities that can especially be supported by the MOSAIC models and thus, we mark them with ✓✓. The last two usage activities cannot be directly supported by MOSAIC and thus, we mark them with o.

Usage Activities		Voting	MosAIC			
			Models ISMs ICM SFM	Analysis activities		
				Selection of ISM practices based on SFM situationalFactors	Identification of similar ISM practices	Identification of dependencies between ISM practices
Identify the process profile of the organization		88%	✓	✓	-	-
Identify the value of reference PRs	Value of reference PRs	25%	✓	✓	-	-
	ROI of reference PRs	0%	✓	✓	-	-
	Additional value of reference PRs	13%	✓	✓	✓	-
	Lost value of reference PRs	13%	✓	✓	✓	-
Identify a new PR		0%	✓✓	-	-	-
Create a tailoring instrument for software projects		50%	✓	✓	-	-
Create a repository with multiple PRs		75%	✓✓	✓	✓	✓
Define the internal process PRs without redundancies	Redundancies in the internal process PRs	13%	✓	-	✓	-
	Redundancies between reference PRs	38%	✓	-	✓	-
	Redundancies between process PRs, reference PRs and internal process PR	13%	✓	-	✓	-
Define the internal process PRs without inconsistencies	Consistent definition of highly dependent PRs practices	13%	✓	-	-	✓
	Consistent dependencies	13%	✓	-	-	✓
Provide enough information in the internal process PRs	Additional information from similar PRs	13%	✓	-	✓	-
	Abstract and concrete PRs practices	13%	✓✓	-	✓	-
	Time order of the PRs practices	50%	✓	-	-	✓
	Achieve a common terminology	50%	✓✓	-	-	-
Maintain the internal process PRs		50%	✓	-	-	✓
Achieve compliance to reference PRs		88%	✓	-	✓	-
Perform efficient assessments	Avoid redundancies in the evaluations	13%	✓		✓	-
	Avoid unnecessary evaluations	0%	✓	-	-	✓
Extension: Non-redundant software requirements		13%	o	-	o	-
Extension: Non-conflicting software requirements		0%	o	-	o	-

Table 34: Summary – Usage activities for MOSAIC

For each activity, we do not only describe the MOSAIC applications, but also its limitations. We summarize the most important limitations that can be considered in the future work.

Based on MOSAIC, we can select ISM practices based on the software project context and thus, support organizations making decision for their process improvement. Other analysis activities, such as the computation of the value of a reference PR, its ROI, the additional or lost value of reference PRs are not currently implemented in MOSAIC. Another limitation of MOSAIC is the selection of ISM practices based on the organization's goals or the selection of ISM practices from PRs for other software areas. We argue that the SFM situationalFactors in MOSAIC are also valid for other software areas or that the organization's goals can be characterized by these SFM situationalFactors. However, this fact must be systematically analyzed in the future work.

In MOSAIC, there are various possibilities to identify similar ISM practices. However, based on the needs of the organizations, the similarities and the differences between PRs can be presented differently. Therefore, based on the similarity or coverage metrics, further analysis activities can be implemented in the MOSAIC Toolbox. For example, the MOSAIC Toolbox can be extended to automatically create a mapping or identify the gap between PRs. Finally, the MOSAIC Toolbox can be extended so that abstract and concrete ISM practices are extracted automatically. Therefore, the creation of a repository of multiple PRs or the transformation of existing PRs into new PRs is supported.

With MOSAIC, we can also identify the dependencies between ISM practices. The MOSAIC Toolbox can be extended to automatically analyze PRs. For example, the automated identification of inconsistencies and redundancies in the PRs can support the definition and maintenance of PRs.

The identification of similarities and dependencies based on MOSAIC metrics is time-consuming due to the detailed results that can be automatically delivered. If a high number of ISM practices are considered as input, the duration for the computation is high as it is performed at the fine-grained level of ICM concepts. However, the results are very detailed as the similarity of each two ICM concepts is considered for the identification of similarities and dependencies between ISM practices. Furthermore, an automation of the analysis activities is possible.

Finally, MOSAIC can inspire other activities that are not related to PRs. For example, the identification of redundancies in the software requirements is such a case. The MOSAIC meta-models and models need to be adapted to the new terminology of the target domain in which the requirements are specified.

8 Evaluation

The main goal of this evaluation is to verify the plausibility of the MOSAIC analysis and modeling activities with the support of the MOSAIC Toolbox. An approach is plausible if there is a high correlation between its results and the observations of persons which evaluate this approach [Ludewig and Lichter 2010]. If the modeling and analysis activities are plausible, the goals defined in the first chapter are achieved.

We also aim to verify the usefulness of MOSAIC for organizations working with multiple PRs. Although several experts acknowledged the usefulness of the MOSAIC models and analysis activities (chapter 7), we further applied MOSAIC in an organization. Therefore, we could collect first experiences with its application and thus, evaluate its usefulness for organizations in an effective and efficient adoption of PRs and assessment based on PRs.

Another goal is to demonstrate that MOSAIC is flexible and supports the integration of PRs for software development and other software areas, such software operation or IT governance as well.

Finally, we evaluated the quality in use of the MOSAIC Toolbox by analyzing the quality attributes defined by the ISO/IEC 25010 [ISO/IEC 25010:2011 2011].

8.1 Types of Evaluation

In the first step, we conducted an evaluation of the MOSAIC approach and thus, of its parts: the models, modeling activities, analysis activities and their underlying metrics. This evaluation is supported by the MOSAIC Toolbox as this supported us to create these models and perform these activities. However, this evaluation is not sufficient to evaluate a tool. Consequently, in the second step, we evaluated the MOSAIC Toolbox according to the quality in use model [ISO/IEC 25010:2011 2011].

An approach can be evaluated by different empirical strategies [Wohlin 2012]. There are three types of such empirical evaluation strategies which engage subjects in the activities performed:

- *Experiments (E)* control a situation by defining independent and dependent variables. These variables manipulate the behavior of an approach directly, precisely and systematically. For example, experiments are best suited to evaluate metrics or algorithms [Wohlin 2012].
- *Case studies (CS)* are observational studies and are less controlled as experiments. Data about an approach is collected to analyze this approach. This analysis can mean to track a specific attribute or to establish relationships between different attributes of this approach. Therefore, an evaluation and if necessary an improvement of the approach related to this attribute or to its relationships is performed.
  Field studies (FS) are special case studies. These are strategies to evaluate an approach in the context of organizations [Ludewig and Lichter 2010]. The evaluation is performed in software projects or other parts of one or more organizations. The experiences of the organizations' employees with the approach are collected to evaluate and eventually improve it. The field studies are not as much controlled as the case studies. Therefore, the costs are lower than the costs of the case studies, but the results are not as reliable.
- *Surveys (S)* are retrospective studies for the evaluation of an approach at the end of its development to obtain feedback of the subjects involved.

To evaluate MOSAIC, we performed the following evaluation activities that involved the participation of various experts (Fig. 51):

- *Experiments (E).* We evaluated the plausibility of two analysis activities, i.e. identification of similar ISM practices and of dependencies between them.
- *Case and Field studies (CS and FS).*
  - In two case studies, we evaluated and improved the MOSAIC models, as well as the modeling and analysis activities. We investigated various PRs and the software project context to collect relevant data about how to model and how to relate them. We investigated the modeling and analysis activities in the case study "Building MOSAIC models" and evaluated an analysis activity in particular, namely the selection of ISM practices based on SFM situationalFactors, in the case study "Mapping of ISM practices to SFM situationalFactors".
  - In a field study, we evaluated the usefulness of MOSAIC and partially evaluated the plausibility of the analysis activities.

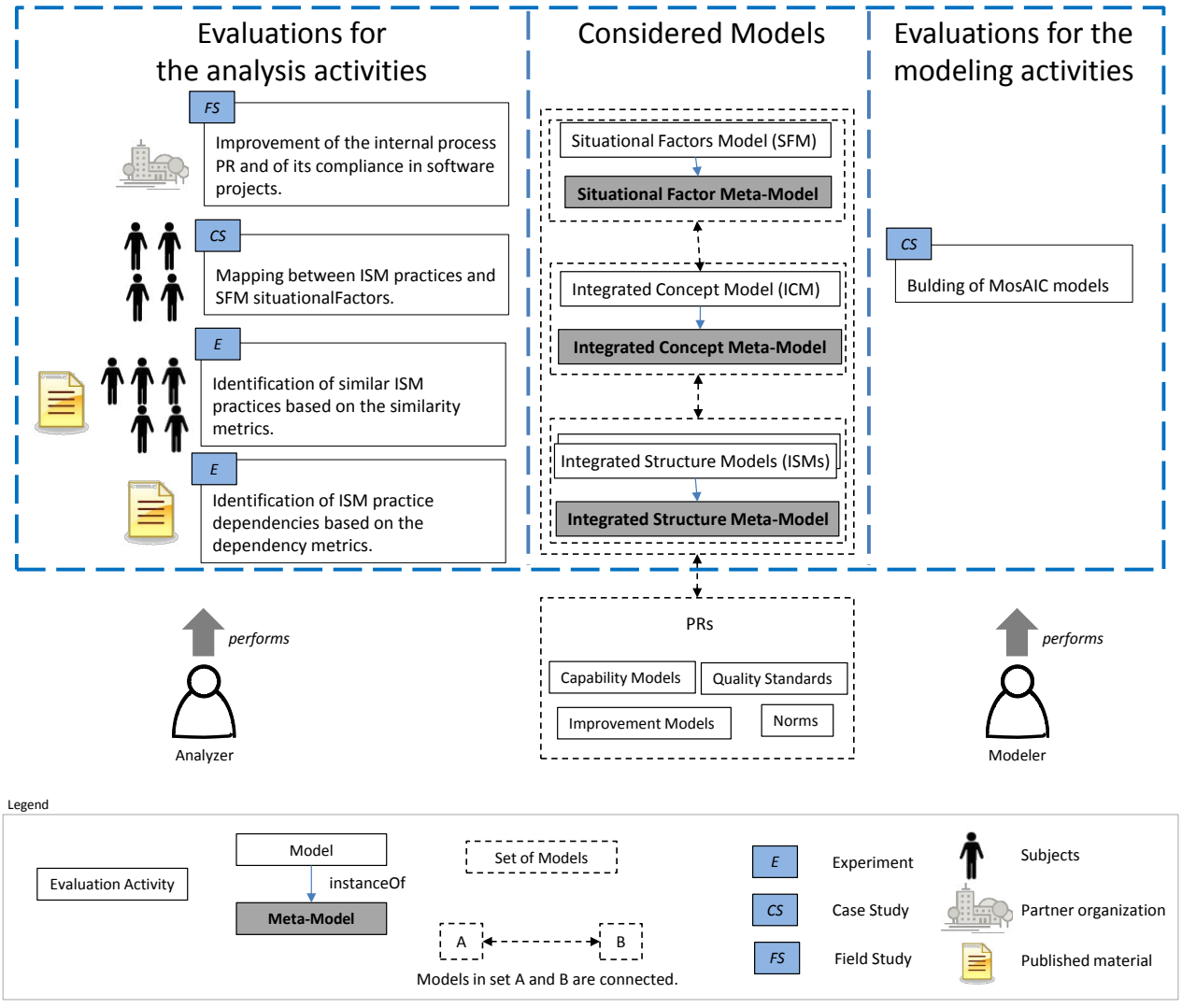


Fig. 51: Evaluation activities for MOSAIC

8.2 Subjects Profile

In this section, we provide an overview of the subjects involved in the evaluation activities of MOSAIC and of the MOSAIC Toolbox (Table 35).

Various experts from different organizations were being engaged as appropriate in the evaluation activities to acquire a broader feedback and to use this expertise to improve MOSAIC. Table 35 lists the different subjects' profiles and their skills: experience with different PRs, number of years' experience in process improvement based on the mentioned PRs, active involvement in process working groups at the international standardization level and experience in different organizations.

Firstly, industrial software consultants from different consultant organizations specialized in the adoption of multiple PRs and assessment based on PRs were involved. They play various organizational roles in the software process improvement initiatives of organizations.

Secondly, experienced academic software process researchers with different academic degrees and from research institutes in different countries were involved. They also play various organizational roles in the software process improvement initiatives of organizations.

Finally, industrial partners from two organizations were involved. These two organizations are both the IT of insurance companies and are located in Germany:

- Organization A has about 1350 employees and is located in more cities (Cologne, Düsseldorf, Munich and Hamburg). It started in 2012 with the process improvement according to CMMI-DEV. This organization supported us intensively in the evaluation of MOSAIC. It uses the MOSAIC ideas and results for its software process improvement program at the project and organizational level. We do not mention all the organizational roles that use the MOSAIC results, but only the ones that intensively participated in the evaluation of MOSAIC. We closely interacted with a software process engineer (Table 35, ID 5) and one process consultant in this organization (Table 35, ID 4).
- Organization B has about 1000 employers and is located in more cities (Aachen, Cologne, Munich and Hamburg). The organization has experiences working with multiple PRs. Their process improvement program started in 2005 with the adoption of CMMI-DEV and they achieved the maturity level 3 in 2012. Furthermore, the organization also considered the adoption of ITIL. Currently, it is interested in the adoption of COBIT and certification according to CMMI-SVC. Two of its software process engineers participated in the evaluation activities of MOSAIC (Table 35, ID 6/7). Four software process engineers and one software process improvement lead supported us to evaluate the MOSAIC Toolbox (Table 35, ID 6-10).

ID	Subjects	Organizational roles	PRs knowledge					Process experience (years)	Process working groups	Multiple organisations
			CMMI-DEV	SPICE	CMMI-SVC	ITIL	COBIT			
1	Consultants	Various organizational roles	✓	✓	✓	✓	✓	> 10	-	✓
2			✓	✓	✓	-	-	> 10	-	✓
3			✓	✓		-	-	10	✓	✓
4			✓	✓	✓	-	-	>10	✓	✓
5	Industrial partners Organization A	Software process engineer	✓	-	-	-	-	3	-	-
6	Industrial partners Organization B	Software process engineer	✓	✓	✓	✓	-	9	-	-
7		Software process engineer	-	-	✓	✓	-	3	-	-
8		Software process engineer	✓	-	-	✓	-	7	-	-
9		Software process engineer	✓	-	-	-	-	9	-	-
10		Software process improvement lead	-	-	✓	✓	-	5	-	-
11	Academic Researcher	Various organizational roles	✓	✓	-	-	-	>10	✓	✓
12			✓	✓	-	-	-	>10	✓	✓
13			✓	✓	✓	✓		10	✓	✓
14			✓	✓	-	-	-	>10	-	✓

Table 35: Evaluation - Overview of the subjects profile

8.3 Experiments

The main goal of the experiments is to evaluate the plausibility of two MOSAIC analysis activities. We performed the following experiments (Fig. 52):

- *Experiment – Identification of similar ISM practices*. We selected ISM practices from multiple PRs and involved subjects to evaluate the similarity degree of their ISM activityUnits.
- *Experiment – Identification of dependencies between ISM practices*. We selected ISM practices from a PR and used the results from published materials to evaluate the dependency degree between its ISM practices.

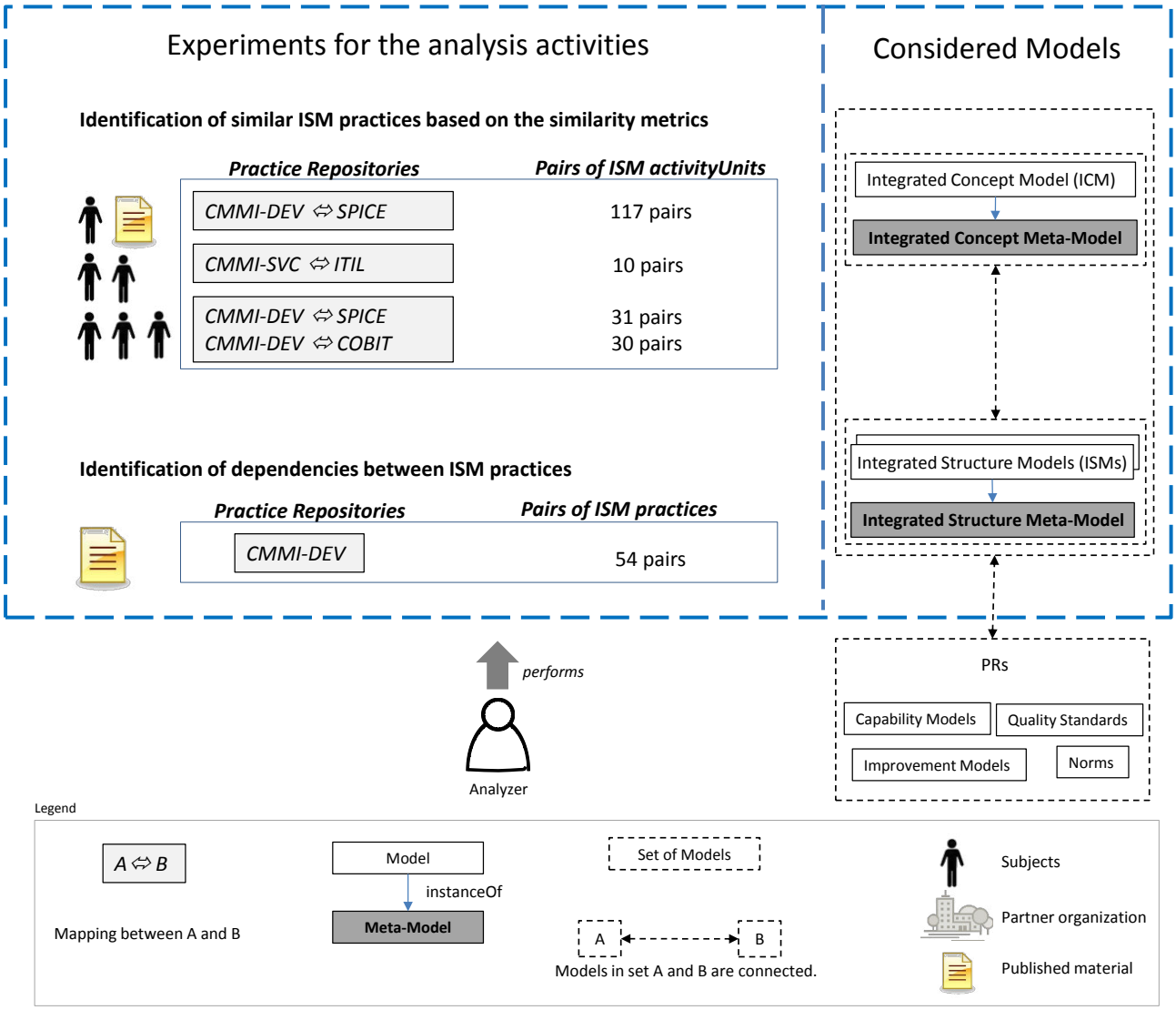


Fig. 52: Evaluation – Experiments

We performed experiments according to a process that contains four main activities: definition, planning, operation, analysis & interpretation [Wohlin 2012].

8.3.1 Definition

In this activity, we defined the goals of the experiments according to the following parameters [Briand et al. 1996]:

- *Object of Study [what is studied]*: The objects of the experiments were the following **analysis activities:**
  - The identification of similar ISM practices based on the similarity metrics. For a more precise evaluation, we evaluated the similarity metrics for ISM activityUnits instead for ISM practices. The reason is that the ISM practices are described differently in the differ-

ent PRs (one can contain a single ISM activityUnit, other can contain several ISM activityUnits). Consequently, the similarity degree between these ISM practices is not very precise.
    - The identification of dependencies between ISM practices based on the dependency metrics
- *Purpose [what is the intention]*: The purpose was the **evaluation** of the experiments objects.
- *Quality Focus [which effect is studied]*: The focus is on the **plausibility** of the experiments' objects.
- *Perspective [whose view]*: The experiments were performed from the **Analyzer** perspective.
- *Context [which environment]*: The context was determined by the **artifacts** used and the **subjects** involved in the experiments:
    - The **artifacts (input data)** used in the experiments had to be ISM practices of different PRs.
    - The **subjects (experts)** involved in our experiments had to be selected according to their experience with the selected PRs, number of years' experience in the process improvement based on these PRs and experience in different organizations.

8.3.2 Planning

Based on the goals defined in the previous activity, we performed the following six steps:

- *Context selection*: The context within the experiments was not a real software project, it was determined by professional and researchers as subjects and real specific data as artifacts.
- *Hypothesis definition*: There is a high correlation between the subjective results (observations) obtained by different subjects and the results of the analysis activities for the selected artifacts.
- *Variables selection:*
    - The **dependent variables** were the two approaches that are used to obtain the analysis activities results. On the one side, the subjects manually had to perform the analysis activities. On the other side, we had to perform the same analysis activities with the support of the MOSAIC Toolbox.
    - The **independent variables** were different subjects and artifacts that represented the context of the experiments and were varied during the experiments.
- *Design*: We planned to perform the following steps:

    1. Identify artifacts and subjects for the experiments.
    2. Perform analysis activities.
    3. Calculate and analyze the deviation between the subjects' results and the MOSAIC results.

- *Validity evaluation*: Although we involved different expertise and we systematically identified a high number of artifacts to cover as many results as possible, there are some threats to the validity of these evaluations. Not only the planning of the experiments, but also the operation of the experiments leads to threats. Therefore, we summarize them at the end of the chapter and propose further work to address these threats.

8.3.3 Operation and Analysis & Interpretation

According to the planned steps, we evaluated the two analysis activities: identification of similar ISM practices based on similarity metrics and the evaluation of the identification of dependencies between ISM practices based on the dependency metrics.

8.3.3.1 Identification of Similar ISM Practices based on Similarity Metrics

Firstly, we identified artifacts to be used in these experiments to cover different values of the similarity degree. Therefore, we identified 188 ISM activityUnits pairs from ISM practices of CMMI-DEV, SPICE, COBIT and ITIL. Furthermore, we identified the subjects according to their experiences with the considered PRs. We distributed the artifacts to these subjects.

There are three modalities to distribute artifacts to subjects: randomization, blocking and balancing design principle [Wohlin 2012]. In our experiments, the artifacts were not randomly assigned to the subjects (randomization design principle), but we built blocks of artifacts based on the PRs' experience of the subjects (blocking design principle). The number of subjects in the different blocks was 2-3 participants and thus, the blocks were partially balanced (balancing design principle). Consequently, we distributed the subjects to PRs as follows:

- *CMMI-DEV, SPICE, COBIT*: We identified two consultants and one industrial partner (Table 35, ID 1/2/6). 61 ISM activityUnit pairs from these PRs were assigned to these three subjects.
- *CMMI-SVC, ITIL*: We identified a consultant and one industrial partner (Table 35, ID 1/7). 10 ISM activityUnit pairs from these PRs were assigned to these two subjects.
- *CMMI-DEV, SPICE*: We identified a consultant (Table 35, ID 3). Moreover, we used the mapping materials from ISCN (International Software Consulting Group) created by another consultant of the ISCM group to evaluate the MOSAIC results. 117 ISM activityUnit pairs from these PRs were assigned to these two subjects.

Secondly, we performed the analysis activities. The subjects individually identified the similarity degree for the ISM activityUnits pairs according to the 5-point ordinal scale with the values "Equal", "High", "Medium", "Low" and "Non-Equal". We also computed the similarity degree with the MOSAIC Toolbox and identified the similarity degree on a ratio scale. To calculate the deviation, we mapped our results to the 5-point ordinal scale as in Table 13 in chapter 5: [1, 1] as "Equal"*;* [0.67, 1) as "High"*;* [0.3, 0.67) as "Medium"*;* (0, 0.3) as "Low"*;* [0,0] as "Non-Equal".

Finally, we calculated the deviation between the subjects' results and the MOSAIC results. We obtained a deviation of 0.25 for the comparison of CMMI-DEV and COBIT, 0.26 for CMMI-DEV and SPICE and 0.0 for CMMI-SVC and ITIL. The deviation is the number of incorrect results (the MOSAIC result scale value is not equal to subjects' result scale value) divided by the number of compared pairs. This deviation indicates that on average less than every fourth metric result deviates from the given scale value.

These results are promising and indicate that there is a high correlation between the results (observations) obtained by different subjects and the MOSAIC results. Therefore, the hypothesis stated in the planning activity was evaluated.

The deviations were mainly caused by missing mappings between ISM practiceConcepts and ICM concepts and missing or inaccurate similarity relations between the ICM concepts. Based on these evaluations, we performed some improvements of the ISM and ICM models.

We also demonstrated that the integration of PRs for other software areas beside the software development is also possible. We integrated COBIT, CMMI-SVC and ITIL and performed the experiments based on their models.

8.3.3.2 Identification of ISM Practices Dependencies based on Dependency Metrics

Firstly, we identified 54 pairs of ISM practices of CMMI-DEV from the following CMMI-DEV processes: REQM (Requirements Management), MA (Measurement and Analysis), CM (Configuration Management), PPQA (Process and Product Quality Assurance) and SAM (Supplier Management) that have dependencies. Furthermore, as we could not engage experts as subjects, we used a published material which documents the dependencies between the CMMI-DEV practices [Chen et al. 2008].

Secondly, we performed the analysis activity. The subjects' results were given by the dependencies for the 54 pairs of ISM practices documented in the published material. The MOSAIC results were obtained with the MOSAIC Toolbox.

Finally, we calculated the deviation between the subjects' results and the MOSAIC results. We obtained a deviation of 0.19 (every fifth MOSAIC result deviates by one point from the experts' result) for the 54 dependencies between the 24 CMMI-DEV practices.

The results are promising and indicate that there is a high correlation between the results (observations) obtained by different subjects and the MOSAIC results. Therefore, the hypothesis stated in the planning activity was evaluated.

A deviation was expected as we did not model ISM artifacts that are not specified in the ISM practice or in its description. The authors of the published materials considered for the identification of dependencies ISM artifacts that are not specified in the ISM practices or in their description in a PR. For example, for the ISM practice CMMI-DEV MA SP1.4 "Specify how measurement data are analyzed and communicated", they considered the ISM outputs "updated measures" and "updated measurement objectives". These outputs are not specified in the ISM practice or in its description. Therefore, the MOSAIC Toolbox did not identify that CMMI-DEV MA SP1.2 "Specify measures to address measurement objectives" is dependent on CMMI-DEV MA SP1.4 "Specify how measurement data are analyzed and communicated".

8.3.4 Threats to Validity and Future work

For each analysis activity, there are several threats to validity that need to be mentioned. We describe these threats and the future work to address them.

8.3.4.1 Identification of Similar ISM practices based on Similarity Metrics

The evaluation of the identification of similar ISM practices from multiple PRs based on the similarity metrics used the results of subjects with different expertise. From a statistical point of view, the number of experts involved was small. Therefore, a broader involvement of subjects and artifacts from more PRs is needed for a profound evaluation.

Furthermore, the plausibility of the identification of similar ISM practices strongly depends on the plausibility of the MOSAIC models (ISM for each PR and ICM). The modeling activities are performed by the author of this thesis. Although the number of considered artifacts for this evaluation is quite high, the selected artifacts did not covered all the entire created MOSAIC models, so that fault results can be obtained in the future. Therefore, subjects have to be involved in the modeling activities or in several reviews of the MOSAIC models.

8.3.4.2 Identification of ISM practice Dependencies based on Dependency Metrics

The evaluation of the identification of ISM practice dependencies based on the dependency metrics used the results of a published material. Although the authors of the published material mention that a high number of experts are involved in the identification of the published dependencies, further subjects are needed to validate this MOSAIC analysis activity. Furthermore, the published material covers only one PR (CMMI-DEV), so that more PRs need to be considered in the future.

8.4 Case and Field Studies

The case and field studies helped us to collect experiences to evaluate and improve MOSAIC. The main goal of the case studies was to evaluate the plausibility of the MOSAIC meta-models, analysis and modeling activities, while the main goal of the field study was to evaluate the usefulness of MOSAIC. A secondary goal of the field study was to evaluate the plausibility of the applied MOSAIC analysis activities. However, we did not focus on this evaluation during the field study. We performed the following case and field studies (Fig. 53):

- *Case study – Building of MOSAIC models*. We extracted over 2000 MOSAIC elements from CMMI-DEV, CMMI-SVC, SPICE, ITIL, COBIT and internal process PR of one organization, as well as eight elements from the situational factors framework [Clarke and O'Connor 2012] to evaluate and improve the plausibility of the MOSAIC meta-models and of the modeling activities.
- *Case Study – Mapping between ISM practices and SFM situationalFactors*. We performed several mappings between 138 CMMI-DEV practices and eight SFM situationalFactors together with four other experts. We aimed to evaluate and improve the plausibility of the MOSAIC meta-models and of the selection of ISM practices based on SFM situationalFactors.
- *Field Study – Software process improvement in an organization*. We used MOSAIC to support the organization A (section 8.2) to perform a software process improvement according to CMMI-DEV. We created a mapping between 269 internal process PRs practices and 20 SFM situationalFactors and between the 269 internal process PR practices and 138 CMMI-DEV practices. During this field study, we collected first experiences with the application of MOSAIC for the evaluation of its usefulness.

As the field study considered the internal process PR of an organization, we remind about its definition.


Reminder

An **internal process PR** is a special process PR and refers to the internal software processes of an organization. This PR defines activities to be used as guidelines for the improvement of software processes in an organization, e.g. in software projects. Hence, it defines the practices of this organization and thus, it is a PR.

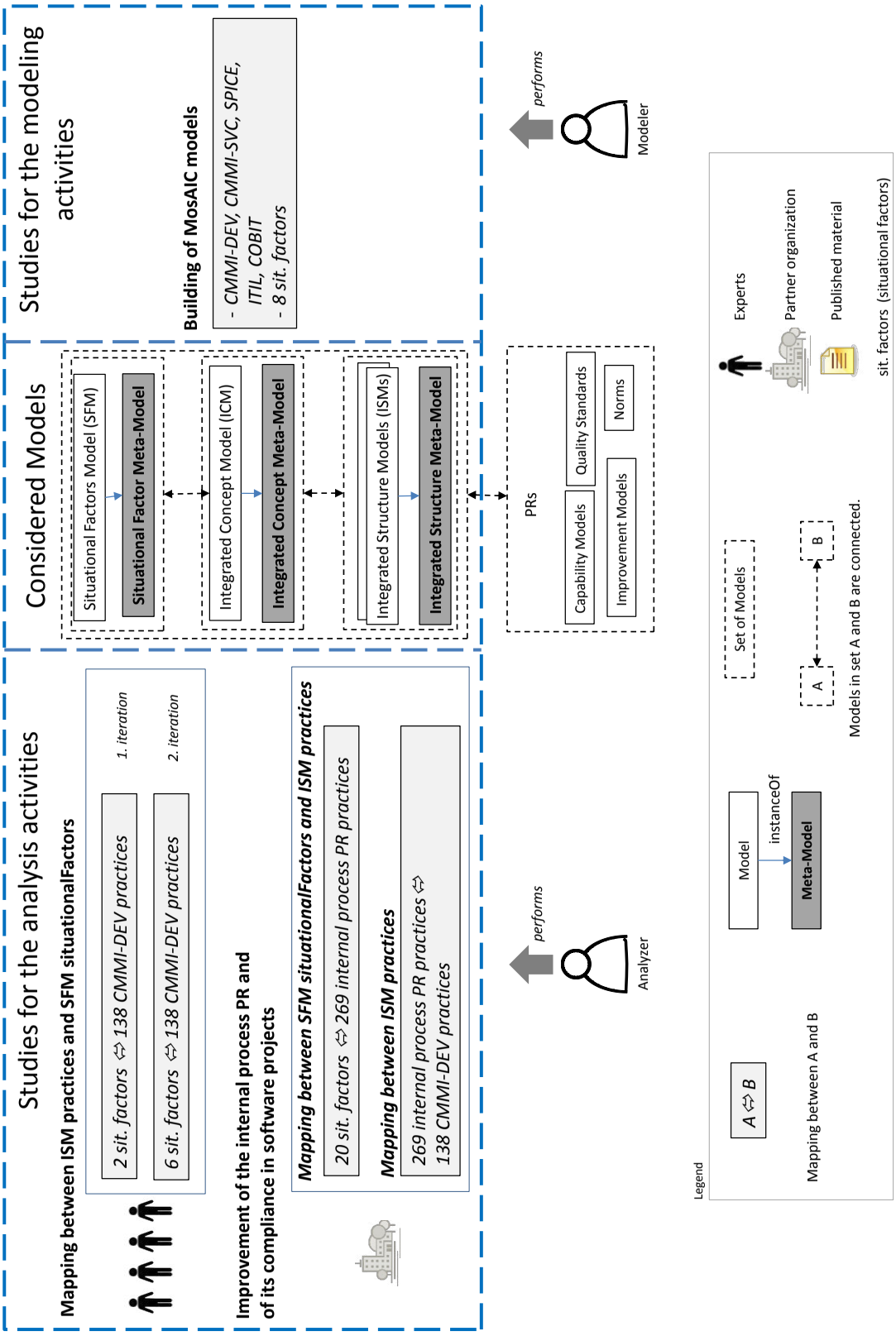


Fig. 53: Evaluation – Case and Field Studies

Analogously to the experiments, we performed the case and field studies according to the process with the four main activities: definition, planning, operation, analysis & interpretation [Wohlin 2012].

8.4.1 Case Study – Building of MOSAIC models

We modeled a high number of MOSAIC elements to evaluate and improve the plausibility of the MOSAIC meta-models and of the modeling activities.

8.4.1.1 Definition

In this activity, we defined the goals of the case study according to the following parameters [Briand et al. 1996]:

- *Object of Study [what is studied]*: The objects of the case study were the MOSAIC meta-models and the modeling activities.
- *Purpose [what is the intention]*: The purpose was the **evaluation** of the objects.
- *Quality Focus [which effect is studied]*: The focus was on the **plausibility** of the objects.
- *Perspective [whose view]*: The case study was performed from the **Modeler** perspective.
- *Context [which environment]*: Different PRs and SFM situationalFactors had to be selected as **artifacts**.

8.4.1.2 Planning

Based on the goals defined in the definition activity, the planning activity was performed according to the following six steps. This activity is similar to the planning activity for experiments except that it does not request the planning of the dependent and independent variables:

- *Context selection*: The context was an off-line, not a real project, without any professional and researchers as subjects, but with real specific data as artifacts. The case study was performed by the author of this work.
- *Hypothesis definition:* We defined the following hypotheses:
    - It is possible to normalize the structure and terminology of multiple PRs and allow an automated identification of similar ISM practices and of dependencies between ISM practices.
    - It is possible to integrate the multiple PRs with the software project context to allow an automated selection of ISM practices.
    - It is possible to integrate PRs for different software areas to allow the automation of the afore-mentioned analysis activities.
- *Design*: We planned to perform the following steps:
    1. Identify PRs and SFM situationalFactors as artifacts for the case study.
    2. Perform the modeling of the selected artifacts with the support of the MOSAIC Toolbox.
    3. Automatically perform analysis activities with the support of the MOSAIC Toolbox.
    4. Analyze the case study and derive lessons learned based on the experiences collected.
- *Validity evaluation:* We analyzed and described the threats to validity and the future work to address these threats.

8.4.1.3 Operation and Analysis & Interpretation

We performed the four steps defined in the planning activity.

8.4.1.3.1 Identify Artifacts

We selected the following PRs as artifacts: CMMI-DEV, SPICE, COBIT, ITIL, CMMI-SVC and the internal process PR of the organization A. We selected these PRs because these cover different software areas: software development, operation and IT governance. Furthermore, the selected reference PRs are commonly used PRs by different organizations [Heston and Phifer 2011]. Finally, we have knowledge and experiences with CMMI-DEV, SPICE and ITIL and thus, we could better understand the structure and terminology used by these PRs. We did not aim to model entire PRs, but only parts of them to evaluate the stated hypotheses (Table 36).

PRs	PRs processes	PRs elements of the PRs processes
CMMI-DEV	CM (Configuration Management)	All SPs (specific practices)
	DAR (Decision Analysis and Resolution)	
	IPM (Integrated Project Management)	
	ISM process	
	MA (Measurement and Analysis)	
	OPD (Organizational Process Definition)	
	OPF (Organizational Process Focus)	
	OT (Organizational Training)	
	PMC (Project Monitoring and Control)	
	PP (Project Planning)	
	PPQA (Process and Product Quality Assurance)	
	REQM (Requirements Management)	
	RSKM (Risk Management)	
	SAM (Supplier Agreement Managemen)	
	PI (Product Integration)	
	RD (Requirements Development)	
	TS (Technical Solution)	
	VAL (Validation)	
	VER (Verification)	
SPICE	SUP.1 (Quality assurance) - only parts	All BPs (best practices) of the processes
	SUP.2 (Verification) - only parts	
	MAN.3 (Project management)	
	ACQ.13 (Project Requirements) - only parts	
	ACQ.15 (Supplier qualification) - only parts	
	SPL.1 (Supplier tendering) - only parts	
	SPL.2 (Product release)	
	ENG.2 (System Requirements Analysis)	
	ENG.3 (System architectural design)	
	ENG.4 (Software Requirements Analysis)	
	ENG.8 (Software testing)	
	ENG.9 (System integration test)	
	ENG.10 (System testing)	
COBIT	PO1 Define a strategic IT Plan	Some control objectives and practices
	PO4 Define the IT Process,Organisation and Relationships	
	PO6 Communicate Management Aims and Direction	
	PO7 Manage IT Human Resources	
	PO8 Customer Focus	
	PO9 Assess and Manage IT Risks	
	PO10 Manage Projects	
ITIL	Incident Managament	All process steps
CMMI-SVC	Incident Resolution and Prevention	All SPs (specifc practices)
Internal process PR	Project management-Scope	All practices
	Project management-Risk management	
	Project management-Communication	
	Quality assurance - Initiation	
	Quality assurance - Planning	
	Quality assurance - Operation	
	Systemdevelopment - Requirements definition	
	Systemdevelopment - Requirements analysis	

Table 36: Case Study – Building MOSAIC Models – Considered PRs' elements

Furthermore, we selected eight SFM situationalFactors to cover different categories from the situational factors framework [Clarke and O'Connor 2012](Table 37).

Category	SFM situationalFactors	Definition
Requirements	Requirements Rigidity (Scope)	Low degree of freedom is permitted in interpreting requirements.
	Requirements Changeability	High volatility of requirements could exist in the project. For example, requirements may be set at the outset of the project or are they may be subject to continuous and extensive changes as the development is underway.
	Requirements Standard	There is a high probability that requirements are missunderstood or there are conflicting requirements or poor quality requirements.
Business	Time to Market	A speed delivery of the product is requested.
Application	Application Performance	There are demands of high performance for the product(s)/application(s) under development. For example, product(s)/application(s) may be required to process a high number of transaction per Second.
Technology	Technologies Emergent	New technology is being used in the development of the software product.
Personnel	Personel Cohesion	There is less experience in working together in the team or the team is highly distributed.
	Personel Disharmony	There exist or could exist conflicts in the team.

Table 37: Case Study – Building MOSAIC Models – Selected SFM situationalFactors

8.4.1.3.2 Modeling of Selected Artifacts

We extracted the ISM and ICM elements from the CMMI-DEV, CMMI-SVC, COBIT and ITIL practices in parallel. We started with these PRs to create an ICM that covers different software areas: software development, software operation and IT-governance. Then we continued with the extraction of elements from SPICE and from the internal process PR as these are PRs for the software development area. Therefore, its ISM practiceConcepts could be related to existing ICM concepts from ICM and thus, the modeling activities were less time-consuming. Then, we extracted the SFM situationalFactors and related them to the ICM concepts.

We mainly used the `XMLModelerTool` to extract the ISM elements, ICM concepts and SFM situationalFactors and related them. Table 38 lists the total number of extracted MOSAIC main elements.

Element		Number
ISM elements	Practices	289
	Outputs	657
	Inputs	394
	Roles	32
	Purposes	56
ICM elements	Concepts	804
SFM elements	SituationalFactors	8

Table 38: Case Study – Building MOSAIC Models – Extracted MOSAIC elements

The ICM with its ICM concepts and similarity relations is the model that connects the PRs with each other and the PRs with the software project context. Therefore, it is the most important model of MOSAIC. To get an impression about the size of the ICM, we present a snapshot of the created

ICM (Fig. 54). The many generalizationOf-mono-hierarchies are represented as stars with their ICM abstract concept in the middle.

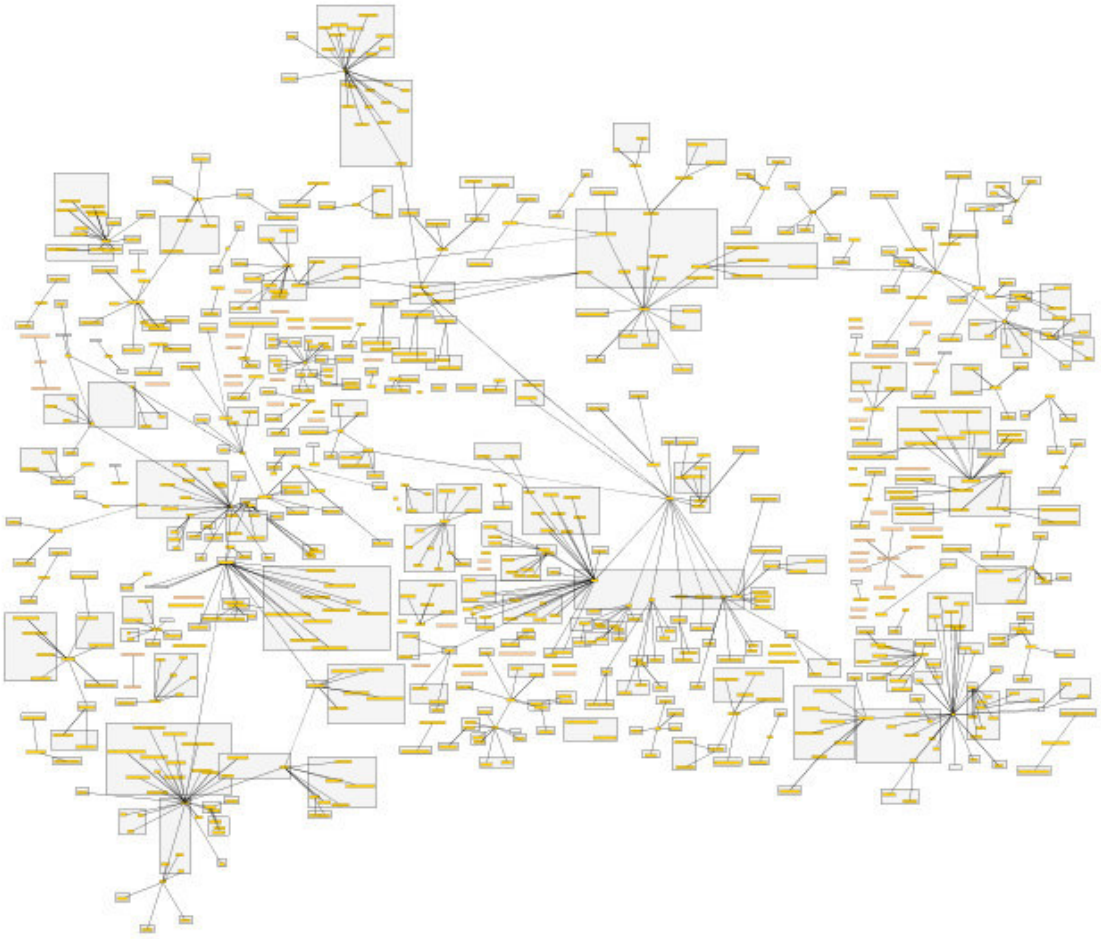


Fig. 54: Case Study – Building MOSAIC Models – ICM

Based on our experiences with the modeling activities, we estimated the effort needed for the modeling of CMMI-DEV level 2 and 3 to give some guidelines for the integration of a PR (Table 39). These estimations are valid for a Modeler that have experience with the MOSAIC modeling activities and have a deep knowledge and understanding of CMMI-DEV, its elements and relations between these elements. If such experiences are not available, then more effort is needed.

We estimated the effort to extract the ISM elements for CMMI-DEV at approximately 7 working hours and to extract the corresponding ICM concepts and relate them to this ISM at approximately 26 working hours. The creation of ICM concepts is costly as a Modeler has to carefully define the similarity relations of the new created ICM concepts.

The effort to relate one SFM situationalFactor to one PR is not relevant, as the SFM situational-Factors are not related to one ISM but to more of them via the ICM concepts. As we modeled parts of CMMI-DEV, SPICE, ITIL, CMMI-SVC, COBIT and internal process PRs and related eight SFM situationalFactors to their ICM concepts, we could perform an estimation based on these experiences.

Therefore, we considered the effort for relating one SFM situationalFactor to all extracted ICM concepts and estimated it at 12 working hours.

Table 39: Case study – Building MOSAIC models - Estimated effort

Considered input elements		No. of elements - Justification	No. of elements	Effort - Element (min)	Effort (min)	Effort (hours)	Total Effort (hours)
Integration of PRs							
CMMI-DEV (practices level 2 and 3)	ISM practice repository elements (SM practiceRepository, categories, processes, practices)	The effort for this extraction is smaller than the effort for the next extraction as these elements are mostly given by the original meta-models of the PRs and thus, it is easy to identify them. - 1 ISM practiceRepository - 5 ISM categories - 18 ISM processes - 138 ISM practices	162	0.3	48.6	0.81	33.26
	ISM practice elements	The effort for this extraction is greater than the effort for the previous extraction as these elements have to be identified in the ISM practices or their description. - 206 ISM activities - 279 ISM outputs - 186 ISM inputs - 21 ISM purposes - 10 ISM roles	702	0.5	351	5.85	
	ICM concepts	This extraction is difficult as it requires a deep understanding of their semantic and of their relations to existing elements in ICM. - 1140 ICM concepts (not unique) are related to ISM practiceConcepts - Some ICM concepts already exist in ICM, some do not. We estimate the extraction and relation of these ICM concepts at 1.4 min: - ICM concept that exist - 0.8 min: - ICM concept that does not exist in ICM - 2 min	1140	1.4	1596	26.6	
Integration PRs with the software project context							
-Entire ICM: over 800 ICM concepts -One SFM situational factor -Four types of relations have to be considered (Concerns, StronglyManagedBy, ManagedBy, Influences)	Relations to ICM abstract concepts (relation valid for all ICM descendant concepts)	Rough estimation for the number of ICM abstract concepts that can be directly related to SFM situationalFactors. - 2/3 from totally 117 ICM abstract concepts are related to SFM situationalFactors	312	0.5	156	2.6	13
	Relations to ICM descendant concepts (relation to ICM abstract concept not valid for all ICM descendant concepts)	Rough estimation for the number of ICM abstract concepts that can not be directly related to SFM situationalFactors. -1/3 from totally 117 ICM abstract concepts are related to SFM situationalFactors - in average 8 ICM descendant concepts for an ICM abstract concept	1248	0.5	624	10.4	

8.4.1.3.3 Perform Automated Analysis Activities

We performed all analysis activities based on the created models. Firstly, the analysis activities with the exception of the identification of similar ISM practices based on the coverage and output state metrics are performed during the evaluations with the different subjects. Secondly, all analysis activities are intensively performed by the author of this work to evaluate their results.

8.4.1.3.4 Lessons Learned

The hypothesis stated in the planning activity was evaluated. First, the normalization of the structure and terminology of the multiple PRs was possible and allowed an automated identification of similar ISM practices and dependencies between them. Furthermore, the integration of PRs from different software areas was also possible. Finally, the integration of multiple PRs and the software project context was possible and allowed the automation of selection of ISM practices based on SFM situationalFactors.

During the case study, we collected experiences with the modeling of PRs. The main lessons learned are:

- The MOSAIC meta-models and modeling activities support a Modeler to build consistent MOSAIC models.
- The building of MOSAIC models requires a high software engineering expertise.
- The building of MOSAIC models requires a high effort that decreases over time.

The MOSAIC meta-models and modeling activities supported us to achieve a robust integration. This integration is mainly based on the ICM and thus, the quality of the integration depends on the quality of the ICM. The structuring of ICM concepts in generalizationOf-mono-hierarchies and restriction of ICM composedOf relations between generalizationOf-mono-hierarchies supported us to create a maintainable ICM. Without these guidelines, the ICM would become too complex to be updated and extended and thus, the integration of multiple PRs with the software project context would not have a high quality.

This case study also revealed that the integration of PRs and of the software project context requires a deep knowledge of the software processes, of the PRs and of the software project context. There are several challenges for these integrations that are worth to be mentioned and carefully handled by a Modeler:

- *Extraction of ISM practices*. While some PRs are well-structured and the ISM practices are easy to identify (e.g. CMMI-DEV, SPICE), other PRs are very detailed and define not only ISM practices, but also additional information how to adopt them or about the meaning of the terms used in such ISM practices (e.g. ITIL). All this non-relevant information needs to be identified as additional information so that the ISM practices can be correctly extracted.
- *Extraction of ISM practice elements and of ICM concepts*. Not all PRs are properly written using a consistent level across all ISM processes. The identification of ISM practice elements requires sometimes an analysis of the ISM practice description for a better understanding of these elements. This is also necessary for the extraction of the ICM concepts based on ISM practiceConcepts because the understanding of what exactly an ICM means is very important for the integration of PRs. Wrong interpretations of the ISM practiceConcepts and their ICM concepts lead to fault results for the analysis activities.
- *Creation and relation of ICM concepts*. The creation of ICM concepts includes the definition of the similarity relations between ICM concepts. These relations need to be carefully defined. As

there is a high number of ICM similar concepts, it is not always easy to differentiate between these similarities and to insert an ICM concept in the right spot in the generalizationOf-mono-hierarchy. This requires a high understanding of the IT context and its terminology. Incorrect definition of the similarity relations between ICM concepts lead to fault results for the analysis activities.
- *Relation of SFM situationalFactors to ICM concepts*. This requires a high experience in software projects to be able to identify the ICM concepts whose adoption support a project to manage a certain critical situation described by a SFM situationalFactor.

We also learned that the building of MOSAIC models requires a high effort. Firstly, the modeling at a very detailed level in MOSAIC is necessary to allow the automation of the analysis activities. However, this is time-consuming. Secondly, the extraction of ISM elements is a challenge. As already mentioned, not all PRs are properly written using a consistent level across all ISM processes. The different structure and the information abundance, the usage of different terminology not only in different PRs, but also within a PR leads to difficult and intensive modeling activities. Finally, the extraction of ICM concepts and relation of these ICM concepts to the ISMs is time-consuming. However, the effort needed for the extraction of ICM concepts and their relation to the ISM decreases if MOSAIC contains a high number of PRs. This is because the ICM becomes stable and less and less ICM concepts are needed to be created and related in the ICM.

8.4.1.4 Threats to Validity and Future Work

The modeling activities were performed by the author of this work and no other experts were involved in the case study. Therefore, there is the threat that any another Modeler might have difficulties to integrate the multiple PRs and the software project context based on the defined meta-models and modeling activities.

The ISMs and ICM were created based on our experiences. Therefore, experts have to be involved in the modeling activities or in reviews of the created ISMs and ICM. The SFM inclusively the relations to the ICM were created based on the results of the case study performed in collaboration with four experts (next section) and thus, the quality of these models is high.

8.4.2 Case Study – Mapping between ISM Practices and SFM Situational-Factors

We performed several mapping exercises together with four other experts to evaluate and improve the plausibility of the MOSAIC meta-models and of the selection of ISM practices based on SFM situationalFactors.

8.4.2.1 Definition

In this activity, we defined the goals of the case study according to the following parameters [Briand et al. 1996]:

- *Object of Study [what is studied]*: The objects of the case study were the MOSAIC meta-models and the analysis activity, the selection of ISM practices based on SFM situationalFactors.
- *Purpose [what is the intention]*: The purpose was the **evaluation** of the objects.
- *Quality Focus [which effect is studied]*: The focus was on the **plausibility** of the objects.
- *Perspective [whose view]*: The case study was performed from the **Analyzer** perspective.

- *Context [which environment]*: PRs and SFM situationalFactors had to be selected as **artifacts**. **Subjects** that have experience with these PRs and with software projects had to be selected.

8.4.2.2 Planning

Based on the goals defined in the definition activity, the planning activity was performed according to the following six steps:

- *Context selection*: The context was not a real software project, it was determined by researchers as subjects and real specific data as artifacts.
- *Hypothesis definition:* We defined the hypotheses:
  - The MOSAIC meta-models define the needed elements to allow an automated selection of ISM practices based on SFM situationalFactors.
  - There is a high correlation between the subjective results (observations) obtained by several experts and the MOSAIC results.
  - It is possible to use the obtained results to relate the software project context with the multiple PRs in MOSAIC.
- *Design*: We planned to perform the following steps:
  1. Identify subjects and artifacts.
  2. Perform a manual mapping between the selected artifacts.
  3. Evaluate and improve the MOSAIC models based on the experiences collected.
  4. Evaluate the MOSAIC selection of the ISM practices based on SFM situationalFactors.
  5. Analyze the case study and derive lessons learned based on the experiences collected.
- *Validity evaluation:* We analyzed and described the threats to validity and the future work to address these threats.

8.4.2.3 Operation and Analysis & Interpretation

We performed the four steps defined in the planning activity.

8.4.2.3.1 Identify Artifacts and Subjects

We invited and motivated different subjects to participate in the case study (Table 35, ID 11-14). Within this case study, we considered eight SFM situationalFactors (Table 37) and all CMMI-DEV practices of level 2 and 3.

8.4.2.3.2 Mapping between CMMI-DEV Practices and SFM SituationalFactors

Together with four subjects, we mapped the CMMI-DEV practices to SFM situationalFactors by defining the support degree of the ISM practices for a software project context characterized by these SFM situationalFactors. These mappings were individually defined during a first iteration and then collaboratively during a second iteration.

The first iteration consisted of the following steps:

1. *Conduct multiple independent mappings between two SFM situationalFactors and CMMI-DEV*. In the first iteration, we considered two SFM situationalFactors ("application performance" and "requirements changeability") and all CMMI-DEV practices of level 2 and 3. Four subjects

individually attempted an initial mapping, with a fifth subject performing the review of the mappings. A template was created to document the subjective mappings of each subject (Table 40 provides a snapshot). This template contains all the afore-mentioned CMMI-DEV practices categorized by their process areas and maturity levels. The support degree between an ISM practice and a SFM situationalFactor was identified according to a four-point ordinal scale (3: Strong, 2: Moderate, 1: Weak, 0: Absent). Each subject specified the support degree by marking the corresponding cell with "x". As the mappings are subjective, we introduced a justification column to document the reasoning of the subject for the chosen degree.

2. *Analyze mappings for deviations and commonalities*. In several discussions, the subjects used the idea of the "poker planning"-method for cost estimation asking the contributors with the minimum and maximum supporting degree to justify their decision. This often led to an adjustment of the initial inputs.

3. *Review the recorded mappings*. The consolidated and analyzed mappings were independently evaluated by a subject who was not involved in the mapping process up to this point. He specified the support degree by marking the corresponding cell with "x" in the template. Furthermore, he identified the frequencies of provided support degrees as a mechanism for taking all views into account. These frequencies also assisted us to calculate the overall support degree between a SFM situationalFactor and the CMMI-DEV practice.

IRM practices / SFM situationalFactor		Required performance of application(s)/product(s)				
		Concerned with the performance demands that are placed product(s)/application(s) under development. For example, product(s)/application(s) may be required to process a high number of transaction per second.				
		Consolidated Independent Mappings				Justification
CMMI Process Area		3: Strong	2: Moderate	1: Weak	0: Absent	
Measurement and Analysis	MA					
Establish and maintain measurement objectives that are derived from identified information needs and objectives.	SP 1.1	x	xxxx			If there are specific performance requirements, then it may be necessary to set objectives and measures in relation to the performance of application(s)/product(s).
Specify measures to address the measurement objectives.	SP 1.2		xxxxx			
Specify how measurement data will be obtained and stored.	SP 1.3			xxxx	x	Although the collection of measurement data may be importnant where performance is an important consideration, this does not imply that it is necessary to specifiy how the measurements will be obtained or analysed.
Specify how measurement data will be analyzed and reported.	SP 1.4			xxx	xx	
Obtain specified measurement data.	SP 2.1		xxxx	x		If ther are specific performance criteria to satisfy, then the collection and analysis of the measurement data is going to be necessary.
Analyze and interpret measurement data.	SP 2.2		xxxxx			
Manage and store measurement data, measurement specifications, and analysis results.	SP 2.3		x	xxxx		If there are specific performance criteria to satisfy, then the measurement data/results may need to be stored and communicated to stakeholders.
Report results of measurement and analysis activities to all relevant stakeholders.	SP 2.4		x	xxx	x	

Table 40: Case Study – Mapping ISM practices, SFM situationalFactors – Excerpt of the used template

The second iteration consisted of the following steps:

1. *Conduct collaborative mappings between six SFM situationalFactors and CMMI-DEV*. We considered six SFM situationalFactors ("requirements standard", "requirements rigidity", "time-to-market", "technology emergency", "personnel disharmony" and "personnel cohesion") to be mapped to all CMMI-DEV practices of level 2 and 3. Four subjects were involved in workshops, with the fifth subject performing a review. Inspired by "poker-planning" cost estimation method, the four subjects used cards with the 4-point ordinal scale values described in the first iteration. For each CMMI-DEV practice and each SFM situationalFactor, the subjects played the cards simultaneously and the subjects with the minimum and maximum supporting degree were asked to justify their decisions. This often led to an adjustment of the initial inputs as the cards were played several times. The results were documented in the template described in the first iteration.

2. *Review the recorded mappings*. As in the first iteration, the mappings were independently evaluated by the subject who was not involved in the mapping process up to this point. He also specified the support degree by marking the corresponding cell with "x" in the template.

Table 41 lists the effort needed to map the CMMI-DEV practices to SFM situational factors. During the case study, we recorded the effort of the subjects involved in the second iteration. As we did not record the effort for the first iteration, we estimated it based on the efforts in the second iteration and chose the maximum effort of mapping one SFM situationalFactor to all CMMI-DEV practices of level 2 and 3 (written in italics). Consequently, each subject spent totally 13.35 working hours to map CMMI-DEV practices of level 2 and 3 to eight SFM situationalFactors and thus, 1.66 hours to map these CMMI-DEV practices to one SFM situationalFactor.

Activities	SFM situationalFactor		Effort (min)	Effort (hours)	Effort (Average hours) One SFM situationalFactor	Total Effort (hours) 8 SFM situationalFactors
	Category	Name				
Independent and collaborative mappings	Requirements	Requirements Rigidity (Scope)	124	2.07	1.66	13.35
		Requirements Changeability	*124*	2.07		
		Requirements Standard	99	1.65		
	Business	Time to Market	116	1.93		
	Application	Application Performance	58	0.97		
	Technology	Technologies Emergent	*124*	2.07		
	Personell	Personel Cohesion	156	2.60		
		Personel Disharmony				

Table 41: Case study – Mapping between ISM practices and SFM situationalFactors – Effort

8.4.2.3.3 Evaluation and Improvement of the MOSAIC Models

The following experiences collected during the mapping exercises supported us to evaluate and improve the MOSAIC meta-models:

- *A definition of SFM situationalFactors is necessary for a common understanding*. The sub-factors defined in the situational factors framework [Clarke and O'Connor 2012] do not always support the understanding of the SFM situationalFactor and thus, they are not enough to describe it. We observed that during the case study, the subjects had different interpretations of a SFM situationalFactor. Consequently, the SF meta-model contains the attribute "definition" for a SFM situationalFactor.
- *A 3-point ordinal scale to define the support degree of an ISM practice is enough*. The subjects had often problems to differentiate between the "Moderate" and "Weak" support degree of a CMMI-DEV practice for a SFM situationalFactor. Therefore, a 3-point is more appropriate instead of a 4-point ordinal scale. This decision was also supported by several software process improvement experts from industry and research. Therefore, the MOSAIC support metrics uses a 3-point ordinal scale.
- *A definition of the semantic of the "Medium" and "Strong" support degree is needed*. The subjects had difficulties to indicate the support degree of an ISM practices for the SFM situationalFactors. Consequently, the SF meta-Model defines the reasoning for a relation (Concerns, StronglyManagedBy, ManagedBy, Influences) to semantically enrich the mapping between an ISM practice and a SFM situationalFactor.

Based on the experiences collected during the mapping exercise and its results, we also extracted the SFM situationalFactors and related them to ICM concepts. This extraction allowed us to improve and evaluate the MOSAIC meta-models and the selection of ISM practices based on SFM situational-Factors.

8.4.2.3.4 Evaluation of the MOSAIC Selection of ISM Practices based on SFM SituationalFactors

Based on the mapping results between CMMI-DEV practices and eight SFM situationalFactors, we evaluated the plausibility of the selection of ISM practices based on SFM situationalFactors. We performed the following steps:

1. *Map subjects' results to MOSAIC results on a 3-point ordinal scale*. The subjects used a 4-point ordinal scale to define the support degree of a CMMI-DEV practice for a SFM situational-Factor. As the MOSAIC support metrics use a 3-point ordinal scale, we mapped the subjects' results to the MOSAIC results as illustrated in the Table 42.

Support degree	
MosAIC	**Case Study**
Strong	Strong
Medium	Medium
	Weak
Absent	Absent

Table 42: Case Study – Mapping between the MOSAIC and subjects' results

2. *Calculate deviation between the two results types*. We calculated the deviation between the subjects' results and MOSAIC results. The deviation is given by the number of incorrect results (the MOSAIC result scale value is not equal to subjects' result scale value) divided by the number of compared pairs. For the eight SFM situationalFactors, we obtained a small deviation of 0.02. This means that on average less than every tenth result deviates from the two-point scale.

3. *Analyze deviations and identify possible improvements*. First, the deviation is small as the selection of ISM practices is based on the MOSAIC models that are build based on the results of the case study. However, together with some subjects, we analyzed this deviation and discovered one main cause. We observed, that the subjects considered that the support degree of an ISM practice is based on the support degree of its ISM outputs, but not on its ISM inputs. MOSAIC also considers the support degree of ISM inputs. To know if this is an issue, a broader evaluation is required.

8.4.2.3.5 Lessons Learned

A challenge of this case study was to assure the participation of suitable expert participants and to assure the coordination issues for its execution. We used a network of personal contacts, which was initially established at international software process conferences. From a starting point of two experts, a third one was recruited. The communication in the first iteration was conducted via email and teleconference facilities (Skype), which was hindered by scheduling/availability of experts, time differences, etc. The second iteration was performed in the same location, so that more and profound discussions were performed that led to a great improvement of MOSAIC.

Therefore, we could evaluate the hypothesis stated in the planning activity. The integration of multiple PRs with the software project context and the selection of ISM practices based on this integration was possible and plausible. A small deviation between the subjects' results and MOSAIC results existed and thus, the results are promising.

Although, we cannot compare the efforts of the mapping between a SFM situationalFactor and the ISM practices with and without MOSAIC (1.66 hours and 13 hours respectively), we can argue that on a long-time perspective, the effort needed with MOSAIC is smaller. These two values cannot be compared due to the following reasons:

- The mapping with MOSAIC considers the relations of SFM situationalFactors to ICM concepts and not to ISM practices directly.
- The estimated effort for a mapping with MOSAIC is valid for a Modeler that have experience with the MOSAIC modeling activities.
- The mapping with MOSAIC is performed based on the experiences gained during the case study and thus, previous knowledge of the relations between SFM situationalFactors and PRs exists.

We argue that the effort needed is smaller as the SFM situationalFactors have to be related only once to ICM concepts. Then, all the various PRs that contain these ICM concepts become related to the SFM situationalFactors. As the various PRs have redundancies, they share a great number of ICM concepts. Consequently, the relation between the SFM situationalFactors and these redundant parts of the various PRs are performed only once.

8.4.2.4 Threats to Validity and Future Work

Different aspects decreased the controllability of this case study and have to be considered as threats to its validity:

- The subjects possible influenced each other as they partially changed their individual results based on the justification or estimations of other subjects.
- Missing definition of the first two SFM situationalFactors led to misinterpretations of the SFM situationalFactors and thus, to misinterpretations of the support degree of a certain ISM practice for this SFM situationalFactor.
- Missing guidelines about when the support degree has a certain value "Strong", "Medium", "Weak", "Absent" led to different interpretations and definition of a mapping between an ISM practice and a SFM situationalFactor.
- The selection of ISM practices based on SFM situationalFactors used the MOSAIC models build based on the results of the case study and thus, the evaluation of the plausibility was restricted.
- Only CMMI-DEV was considered for the mapping between PRs and SFM situationalFactors.

To address these threats, an evaluation has to be performed with subjects that are not involved in the MOSAIC modeling activities. Moreover, fully independent experiments with predefined rules and guidelines are requested. Finally, further PRs have to be considered as input.

8.4.3 Field Study – Software Process Improvement in an Organization

We used MOSAIC to support the organization A (section 8.2) in its software process improvement initiative. This organization aimed to improve its internal process PR for the software development area according to CMMI-DEV and to improve the process compliance in the software projects. This field study allowed us to apply MOSAIC and to evaluate its usefulness.

8.4.3.1 Definition

In this activity, we defined the goals of the field study according to the following parameters [Briand et al. 1996]:

- *Object of Study [what is studied]*: The object of the field study was MOSAIC with the following analysis activities: selection of ISM practices based on SFM situationalFactors and the identification of similar ISM practices based on the similarity and coverage metrics.
- *Purpose [what is the intention]*: The purpose was the **evaluation** of the objects.
- *Quality Focus [which effect is studied]*: The focus was on the **usefulness** of the objects in an organization. The **plausibility** of the objects was also considered, but it was not in focus.
- *Perspective [whose view]*: The field study was performed from the **Analyzer** perspective.
- *Context [which environment]*: Different PRs and SFM situationalFactors had to be selected as **artifacts**. An organization that started the improvement of an internal process PR according to these PRs had to be selected to allow the involvement of **subjects.** Organizations with higher experience in the process improvement according to certain PRs were not suitable for the selection as in general there is no need for the MOSAIC analysis activities anymore when the software process improvement is on progress as these are already performed.

8.4.3.2 Planning

Based on the goals defined in the definition activity, the planning activity was performed according to the following six steps. This activity is similar to the planning activity for case studies:

- *Context selection*: The context consists of a real software process improvement project and several software projects, with professionals as subjects and real specific data as artifacts.
- *Hypothesis definition:* We defined two hypotheses:
    - The application of MOSAIC with the afore-mentioned analysis activities is possible and useful.
    - There is a high correlation between the subjective results (observations) obtained by different subjects and the MOSAIC results.
- *Design:* We planned to perform the following steps:

    1. Identify subjects and artifacts.

    2. Apply MOSAIC to create work products that can be used for the software process improvement.

    3. Analyze the field study and derive lessons learned based on the experiences collected.

- *Validity evaluation:* We analyzed and described the threats to validity and the future work to address these threats.

8.4.3.3 Operation and Analysis & Interpretation

We performed the steps defined in the planning activity.

8.4.3.3.1 Identify Subjects and Artifacts

We identified an organization that starts the software process improvement according to CMMI-DEV. Because of their little experience with CMMI-DEV, we could apply MOSAIC and evaluate its usefulness.

Various Analyzers were involved in the field study: one software process engineer of this organization (Table 35 ID 5), one software process consultant that support the organization in its software process improvement (Table 35 ID 4) and several software project members of three software projects that had approximately 1200, 1500 and 2000 person days per year. These three medium-size software projects were selected to participate in the start phase of the process improvement projects.

We also identified the SFM situationalFactors that were relevant for this organization (Table 43). 16 SFM situationalFactors from the framework of Clarke et al. were identified as relevant. Furthermore, four SFM situationalFactors were defined based on the organizational context.

Category	SFM situationalFactors	Framework (Clarke et al.)
Personnel	Personnel Cohesion	✓
	Personnel Disharmony	✓
	Staffing	-
	Knowledge	✓
Business	External Dependencies	✓
	Customer satisfaction	✓
	Time	✓
	Budget	-
	Appropiate Budget	-
Requirements	Requirements Rigidity	✓
	Requirements Volatility	✓
	Requirements Standard	✓
	Application quality	✓
Application	Performance	✓
	Degree of risks	✓
	Security	-
	Complexity	✓
	Reuse	✓
	Usability	✓
Technology	Technologies Emergent	✓

Table 43: Field study – SFM situationalFactors

8.4.3.3.2 Apply MOSAIC

We performed the following activities where we could demonstrate the application of MOSAIC:

1. *Create a tailoring instrument for software projects*: We created a tailoring instrument for the software projects to recommend internal process PR practices that are best suited for these projects.

2. *Identify the process profile of the organization*: Based on the organizational context defined by the needs and wishes of the software projects, we identified CMMI-DEV practices to be used as reference for the improvement of the internal process PR.

3. *Achieve compliance to reference PRs*: We identified the gaps between the internal PR and CMMI-DEV to improve the internal process PR and be compliant to CMMI-DEV.

4. *Create a repository of multiple PRs and perform efficient assessments*: We created a repository that contains unique ISM practices with references to the internal process PR and CMMI-DEV to evaluate and improve the process compliance in the software projects according to these two PRs.

5. *Provide helpful information for adoption*: We identified similar ISM practices from CMMI-DEV to extract additional information for the adoption of the internal process PR practices that are selected with the tailoring instrument.

To perform these activities, we used the MOSAIC ideas and partially used the MOSAIC Toolbox. For a complete usage of the MOSAIC Toolbox, all the ISM practices of the internal process PR had to be modeled. Due to the high number of ISM practices and due to the restricted time to deliver the results, we only modeled some of its ISM processes (Table 36).

In the following, we give details about each activity.

8.4.3.3.2.1 Create a Tailoring Instrument for Software Projects

We created a tailoring instrument to recommend internal process PR practices to software projects that are best suited for their context. We aimed that the selection of these best suited ISM practices motivates the software project members to adopt these ISM practices and thus, that the process compliance will improve.

Based on the MOSAIC models and on the selection of ISM practices based on SFM situationalFactors, we created such a tailoring instrument. This tailoring instrument contains the following artifacts:

- List of SFM situationalFactors to analyze and evaluate the software project context
- Mapping between the SFM situationalFactors and the internal process PR practices to select the best suited ISM practices based on the analyzed software project context

In the list of SFM situationalFactors, we documented for each SFM situationalFactors its definition (Table 44). This definition describes critical situations that exist or could appear in the software project. To describe if these critical situations are relevant for a software project, we defined the **criticality degree** for a SFM situationalFactor. We used a 3-point ordinal scale to analyze together with the software project members their project context and to evaluate the criticality degree:

- "**Non-Critical**": The situations described by the SFM situationalFactor are not relevant or do not apply for the software project.
- "**Apply**": The situations described by the SFM situationalFactor apply for the software project.
- "**Critical**": The situations described by the SFM situationalFactor apply for the software project, are critical and can jeopardize the achievement of the software project goals.

Category	SFM situationalFactors	Definition (Critical)	Project (Non-Critical, Apply, Critical)
Personnel	Personnel Cohesion	There is less experience in working together in the team or the team is highly distributed.	Non-Critical
	Personnel Disharmony	There exist or could exist conflicts in the team.	Non-Critical
	Staffing	Not all the important roles in the project could be staffed.	Non-Critical
	Knowledge	The project has a high number of young or new employees with little experience.	Critical
Business	External Dependencies	The number of project stakeholders is high and the their involvment is complex.	Critical
	Customer satisfaction	There are particularly high expectations with regard to customer satisfaction. For example, the relationship with a customer is particularly critical because he was disappointed several times in the past by the project results.	Non-Critical
	Time	The deadlines have to be met, delays are "impossible" (e.g. implementation of legislative changes) or lead to great customer dissatisfaction.	Non-Critical
	Budget	The budget compliance is critical.	Critical
	Appropiate Budget	The budget does not fit the project scope and there is a significant deficit between the approved and needed budget.	Non-Critical
Requirements	Requirements Rigidity	Low degree of freedom is permitted in interpreting requirements.	Apply
	Requirements Volatility	High volatility of requirements could exist in the project. For example, requirements may be set at the outset of the project or are they may be subject to continuous and extensive changes as the development is underway.	Apply
	Requirements Standard	There is a high probability that requirements are missunderstood or there are conflicting requirements or poor quality requirements.	Apply
	Application quality	There are high quality requirements (e.g. because it is a particularly critical system with high availability requirements).	Critical
Application	Performance	There are demands of high performance for the product(s)/application(s) under development. For example, product(s)/application(s) may be required to process a high number of transaction peRSecond.	Apply
	Degree of risks	There are high risks in the software project.	Apply
	Security	There are sensitive data used that must be specially protected, eg. customer data for testing purposes.	Non-Critical
	Complexity	The complexity of the architecture, of the business process or the needed hardware is high.	Non-Critical
	Reuse	There are high demands on reusability or on flexible and extensible architecture.	Non-Critical
	Usability	The system to be developed should have a high degree of usability.	Non-Critical
Technology	Technologies Emergent	New technology is being used in the development of the software product.	Non-Critical

Table 44: Field study – Tailoring instrument – Examples of critical situations in a software project

The tailoring instrument also contains a mapping between the internal process PR practices and the SFM situationalFactors. To create this mapping, we computed the support degree of an ISM practice for each SFM situationalFactor. For each internal process PR practice in the tailoring instrument,

we documented only the SFM situationalFactors for which the support degree is “Strong” or “Medium” (Table 45).

ISM Practice	Internal process PR	CMMI-DEV	SFM situationalFactor
	ID	ID	Support Degree
Traceability between requirements, implementation modules and test cases are established.	TS 4.3.2	REQM SP1.4	**Strong:** - Requirements changeability - Requirements rigidity
The project stakeholders are identified and analyzed.	PM 6.1.1	PP SP2.6 IPM SP2.1 GP2.7 (all PAs)	**Strong:** - External dependencies
Project handbook is created.	PM 1.1	PP SP3.3	**Strong:** - Requirements rigidity - Customer satifaction **Medium:** - Time, Budget constraint

Table 45: Field study – Excerpt of the tailoring instrument – Mapping between ISM practice and SFM situationalFactors

To achieve a better quality of these mappings, we validated and improved them as follows. First, we computed the support degree between the CMMI-DEV practices and each SFM situationalFactor. Based on a mapping between the internal process PR and CMMI-DEV, we could use these result for the validation as we assumed that the support degree of ”Equal” and “High” ISM practices is equal. This validation was valuable as it allowed us to use the expertise of five experts that mapped the CMMI-DEV practices to SFM situationalFactors (section 8.4.2). Second, the mappings were individually reviewed by the software process consultant and the software process engineer involved in this field study. The reviewers mostly reported missing mappings and some fault mappings.

Based on this mapping, best suited internal process PR practices for the software projects could be selected. For each SFM situationalFactor that was evaluated as “Critical” for a software project, ISM practices with the support degree “Strong” and “Medium” were selected for this project. For example, the support degree of the ISM practice “The project stakeholders are identified and analyzed” for the SFM situationalFactor “external dependencies” was “Strong”. The criticality degree for this SFM situationalFactor was evaluated as “Critical” for a software project and thus, this ISM practices was selected as best suited practice for this project (Table 46).

ISM Practice	Internal process PR	CMMI-DEV	SFM situationalFactor		Project Tailoring Decision
	ID	ID	Support Degree	Criticality Degree	
Traceability between requirements, implementation modules and test cases are established.	TS 4.3.2	REQM SP1.4	**Strong:** - Requirements changeability - Requirements rigidity	Apply	No
The project stakeholders are identified and analyzed.	*PM 6.1.1*	*PP SP2.6* *IPM SP2.1* *GP2.7 (all PAs)*	***Strong:*** *- External dependencies*	*Critical*	*Yes*
Project handbook is created.	PM 1.1	PP SP3.3	**Strong:** - Requirements rigidity - Customer satifaction **Medium:** - Time, Budget constraint	Apply	Yes

Table 46: Field study – Excerpt of the tailoring instrument – Tailoring decision

8.4.3.3.2.2 Identify the Process Profile of the Organization

Based on the organizational context defined by the needs and wishes of software projects, we identified CMMI-DEV practices to be used as a reference for the improvement of the internal process PR.

Firstly, the context of three software projects was analyzed and the criticality degree of each SFM situationalFactor was evaluated. Then, we identified the most frequent SFM situationalFactors with the criticality degree "Critical" and "Apply". For example, all three software projects mentioned that the external dependencies are very important and complex and thus, the "external dependencies" was identified as one of the most frequent SFM situationalFactor.

Secondly, we used the mapping between CMMI-DEV practices and SFM situationalFactors to select the CMMI-DEV practices for the most frequent SFM situationalFactors. According to the criticality degree of the SFM situationalFactors and support degree of the selected ISM practices, we performed a prioritization for the adoption of the selected CMMI-DEV practices. If the selected ISM practices were already adopted, a review and improvement of the ISM practices (inclusively templates or examples attached to these ISM practices), was performed. For example, the CMMI-DEV PP SP2.6 practices "Plan the stakeholder involvement" was selected as an important ISM practice based on the SFM situationalFactor "external dependencies".

8.4.3.3.2.3 Achieve Compliance to Reference PRs

We identified the gaps between the internal PR and CMMI-DEV to improve the internal process PR and be compliant to CMMI-DEV.

First, we identified "Equal" and "High" similar ISM practices from the internal process PR and CMMI-DEV. Then, for each CMMI-DEV practice that has no "Equal" internal process PR practices, we computed the coverage degree. The coverage degree helped us to verify if the internal process PR practices cover this CMMI-DEV practice.

Based on the similarity and coverage degree, we used a 3-point ordinal scale to evaluate the gap between the CMMI-DEV and internal process PR practices (Table 47).

- **"Red (r)"**: There are no "Equal" or "High" internal process PR practices for the CMMI-DEV practice.

- **"Yellow (y)"**: There are "High" internal process PR practices for the CMMI-DEV practice and the coverage degree is greater than 0.3 and smaller than 1. This means that the "High" internal process PR practices do not entirely cover the CMMI-DEV practice.
- **"Green (g)"**: There are "Equal" internal process PR practices and the coverage degree is 1. This means that the internal process PR practices entirely cover the CMMI-DEV practice.

CMMI-DEV			Gap	Internal process PR	
Maturity Level	ISM process and practice			ISM practice	
	Name	Id		Name	Id
Level 2	**Configuration Management**	**CM**			
	Identify configuration items, components, and related work products to be placed under configuration management.	SP 1.1	r	Use the standard configuration tool to manage the data.	Support-2.4
				Establish a configuration management plan with the identification of configuration items.	Support-2.5
Level 2	Establish and maintain a configuration management and change management system for controlling work products	SP 1.2	y	Use the standard configuration tool to manage the data.	Support-2.6
				Use the change management approach in the project.	Support-2.2.1
				Use the standard configuration tool to manage the changes of the project documents.	Support-2.2.2

Table 47: Field study – Excerpt of the mapping between CMMI-DEV and internal process PR

Based on the evaluation of the "Red (r)" and "Yellow (y)" gap, we identified the differences between the CMMI-DEV and the internal process PR. For example, the gap for the CMMI-DEV CM SP1.2 "Establish and maintain a configuration management and change system for controlling work products" was "Yellow (y)" as the change management approach defined in the internal process PR only addressed project management changes (e.g. time, scope, budget changes) or changes to the project documents, but not technical changes in the development environment. There is no configuration management system that helps the projects to manage the changes across all software development phases.

The identified mappings between CMMI-DEV and the internal process PR were individually reviewed by the software process consultant involved in this field study. He mostly reported missing mappings.

8.4.3.3.2.4 Create a Repository of Multiple PRs and perform Efficient Assessments

We created a repository that contains unique ISM practices with references to the internal process PR and CMMI-DEV. We used this repository to evaluate and improve the process compliance in the software projects according to these two PRs. To evaluate the process compliance according to the internal process PR, we only evaluated the adoption of the selected internal process PR practices during the tailoring. To identify the CMMI-DEV process compliance, we evaluated all the CMMI-DEV practices.

The created repository contains unique ISM practices that cover the internal process PR and CMMI-DEV. As the terminology of the internal process PR is known in the organization, we defined

the unique ISM practices based on the internal process PR practices. However, as the internal process PR did not cover the CMMI-DEV, we also had to define unique ISM practices based on the gap between the CMMI-DEV and the internal process PR.

The repository contains the following three artifacts:

- A list with the unique ISM practices derived from the internal process PR practices
- An invert list with the ISM practices derived from the CMMI-DEV practices
- Process compliance metrics in the software projects

First, the list with unique ISM practices contains the references to the internal process PR and CMMI-DEV practices. For each such unique ISM practice, we documented the IDs of the "Equal" and "High" internal process PR and CMMI-DEV practices. These references to these two PRs allow the identification of further information regarding a certain ISM practice in the PRs description (Table 48). The identified mappings were individually reviewed by the software consultant involved in this field study. As already mentioned, he mostly reported missing mappings and some few fault mappings.

Furthermore, the list with the unique ISM practices contains their corresponding process compliances. We defined the process compliance to evaluate the adoption of the internal process PR and of CMMI-DEV practices in the software projects. For this purpose, we used a 5-point ordinal scale:

- **"Red (r)"**: the adoption is not or not properly performed.
- **"Yellow (y)"**: the adoption is partially performed.
- **"Green (g)"**: the adoption is properly performed.
- **"Not applicable (n.a.)"**: the adoption cannot be performed as it is not applicable or the corresponding ISM practice is not selected for adoption.
- **"Not yet (n.y.)"**: the adoption is not started yet.

ISM Practice	Internal process PR		CMMI-DEV	
	ID	Process compliance	ID	Process compliance
Requirements specification is created.	SE 1.3	r	RD 1.1 RD SP2.1	g
Measurement objectives are systematically identified based on information needs.	-	-	MA SP1.1	r
The metrics SPI, CPI are obtained and analysed.	PM 2.3	g	MA SP2.1 MA SP2.2 PMC 1.1	r
Stakeholder are informed over the project status.	PM 3.4	g	PP GP2.7 PMC GP2.7 IPM GP2.7 PP GP2.10	g

Table 48: Field study – Unique ISM practices that cover the internal process PR and CMMI-DEV

As for each unique ISM practice, there is a single internal process PR practice, the process compliance for the internal process PR could be determined. However, for each unique ISM practice, there are more CMMI-DEV practices. Therefore, the process compliance could not be determined and we needed to create an invert list.

The invert list contains ISM practices derived from the CMMI-DEV practices. For each CMMI-DEV practice, we automatically imported the process compliance of the corresponding unique ISM practices. Based on the process compliance of these unique ISM practices, we manually defined the process compliance for a CMMI-DEV practice (Table 49).

CMMI-DEV				References to the first list	
Maturity Level	ISM process and practice		Process compliance	CMMI-DEV Process compliance	ISM practice
	Name	ID			
	Configuration Management	**CM**			
Level 2	Identify configuration items, components, and related work products to be placed under configuration management.	SP 1.1	r	g	Use the standard configuration tool to manage the data.
				r	Establish a configuration management plan with the identification of configuration items.
Level 2	Establish and maintain a configuration management and change management system for controlling work products	SP 1.2	r	g	Use the standard configuration tool to manage the data.
				r	Use the change management approach in the project.
				r	Use the standard configuration tool to manage the changes of the project documents.
				r	Use the standard configuration and change management tool to controll the work products.

Table 49: Field study – Invert list with CMMI-DEV practices and their compliance

Finally, we defined some metrics to get an overview and monitor the development of the process compliance according to the two PRs in the software projects.

We developed a process compliance matrix which visualizes the process compliances of all ISM practices of a single PR to get an overview of the adoption in a software project. Each column in this matrix refers to a process within the internal PR or CMMI-DEV (Fig. 55).

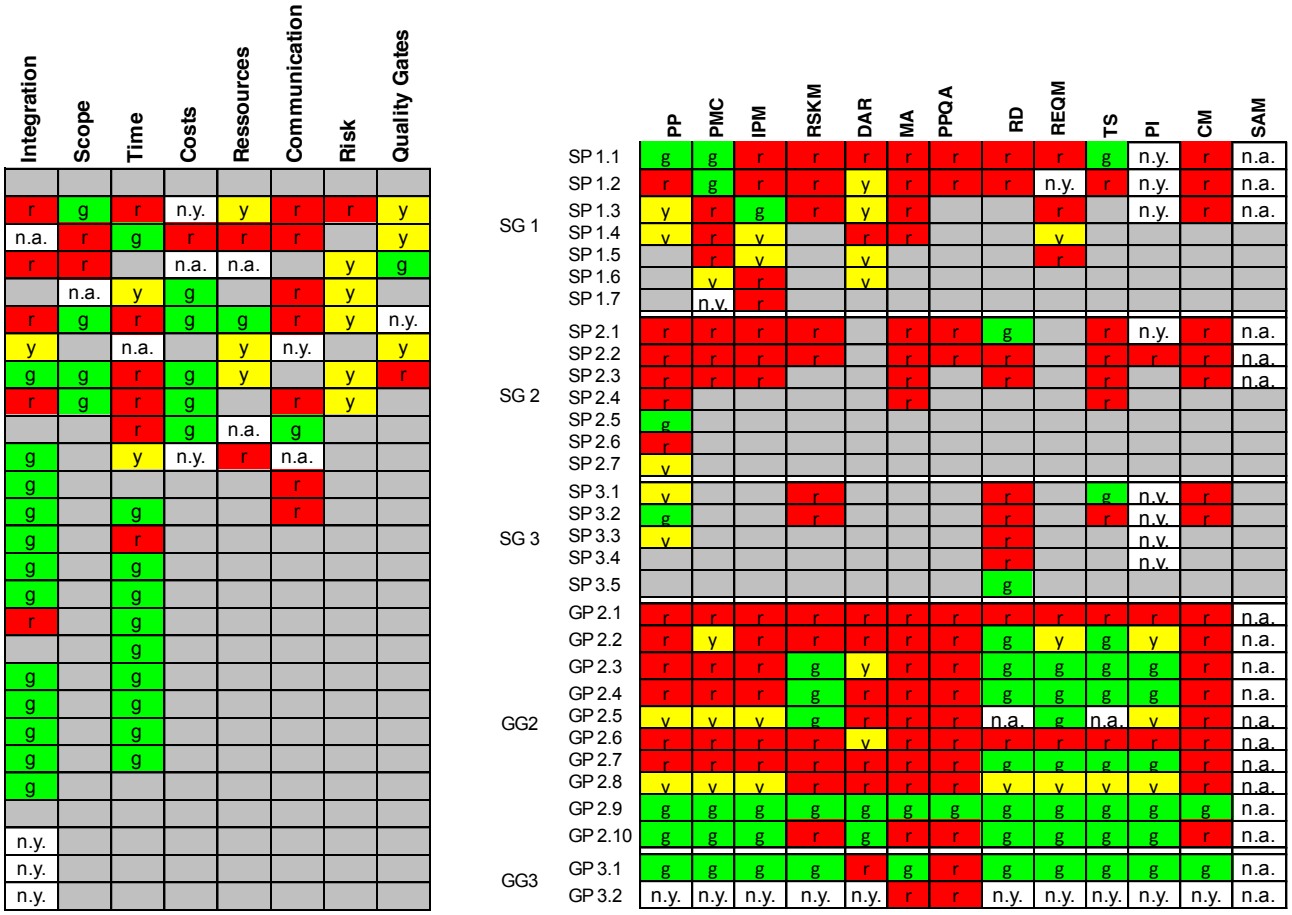


Fig. 55: Field study – Process compliance matrix for the internal process PR and CMMI-DEV

Based on the process compliance matrix, we defined a metric to measure the process compliance and monitored the development of the adoption of each PR. It uses a ratio scale with values between 0 and 1. This metric determines the percentage of ISM practices that are properly adopted by the software project.

$$ProcessCompliance_{PR} = (|Green_{PR}| + 0.5 \cdot Yellow_{PR})$$

To monitor its development, we built a line chart and documented the process compliance for each month when an assessment was performed.

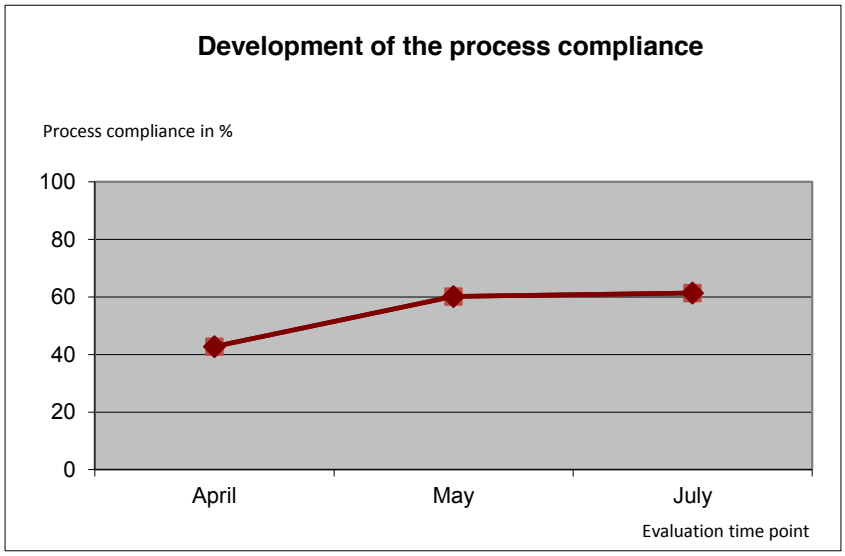


Fig. 56: Field study – Development of the process compliance over time

To summarize, we created a repository with unique ISM practices, their references to the internal process PR and CMMI-DEV practices and their corresponding process compliance in software projects. The repository also contains several metrics to evaluate and monitor the process compliance according to the internal process PR and CMMI-DEV in the software projects.

8.4.3.3.2.5 Provide Helpful Information for Adoption

For some internal process PR practices that were selected for the adoption, the software project members and the software process engineers asked about additional information. The purpose of this additional information was to help them to better understand how to adopt this ISM practice. Consequently, we identified similar CMMI-DEV practices that provide such helpful information for the adoption. For an internal process PR practice, "Equal", "High", "Medium" or "Low" similar CMMI-DEV practices are potential sources to identify this additional information. The PRs description for "Equal" or "High" practices can reveal helpful information. Furthermore "Medium" and "Low" practices have a lower similarity as they have differences. These can be analyzed for relevant information.

For example, for the internal process PR practice "The project stakeholders are identified and analyzed", we identified additional information in the description of the CMMI-DEV PP SP2.6 practice "Plan the involvement of the stakeholders": "A two-dimensional matrix with stakeholders along one axis and project activities along the other axis is a convenient format (…). Relevance of the stakeholder to the activity in a particular project phase and the amount of interaction expected would be shown at the intersection of the project phase activity axis and the stakeholder axis".

8.4.3.3.3 Lessons Learned

The two hypotheses stated in the planning activity were evaluated during this field study.

Firstly, the application of MOSAIC is useful and supports the organization in its software process improvement at the project and organizational level.

At the project level, we created a tailoring instrument and supported the selection of internal process PR practices (section 8.4.3.3.2.1). Only best suited ISM practices were selected for the adoption. The software project members accepted our recommendations as valuable information. In some cases,

they reported that the adoption of some ISM practices has been shown to be beneficial after a period of time. Furthermore, we created a repository with unique ISM practices that cover the two PRs. Redundancies were avoided in the adoption and assessment as similar ISM practice from the internal PR and CMMI-DEV were adopted and assessed only once based on the created repository with unique ISM practices (section 8.4.3.3.2.4). The avoidance of redundancies was confirmed by the software project members. Finally, the software project members found that the provided additional information was helpful to better understand and adopt the ISM practices in their projects (section 8.4.3.3.2.5).

At the organizational level, we identified the gaps between the internal PR and CMMI-DEV. Based on these gaps, we established together with the organization an improvement plan to achieve a compliance to CMMI-DEV in several increments (section 8.4.3.3.2.3). Furthermore, the software process engineers found the additional information from CMMI-DEV to be beneficial and considered it for the description of ISM practices in the internal process PR (section 8.4.3.3.2.5). Finally, we selected CMMI-DEV practices based on the most frequent SFM situationalFactors that could be used as references for the improvement (section 8.4.3.3.2.2). Although the approach for selecting practices at the organizational level was considered as valuable, its results were not considered as input for the process improvement. One reason was that the number of the considered software projects was too small.

Secondly, the application of the MOSAIC analysis activities led to plausible results. All the mappings between the internal process PR and CMMI-DEV practices and between the internal process PR practices and SFM situationalFactors were created. We created them by applying the MOSAIC ideas or directly by using the MOSAIC Toolbox. Therefore, we could observe that the MOSAIC analysis activities are robust. Furthermore, the software project members confirmed and accepted the defined mappings. Finally, the reviewers reported some few fault mappings. An analysis of these fault mappings indicated that these were caused by our faulty interpretations of the ISM practices and not by a possible ineffectiveness of MOSAIC analysis activities.

8.4.3.4 Threats to Validity and Future Work

There are some threats to validity concerning the plausibility and the usefulness of MOSAIC application that need to be mentioned.

Regarding the application of MOSAIC and its usefulness, there are some threats to validity. One threat is given by the small number of software projects that were involved in the field study. For a profound evaluation, it is necessary to apply MOSAIC in more software projects. Furthermore, an evaluation in different organizations is also needed. Finally, we could not evaluate the usefulness of the entire MOSAIC. For example, the identification of dependencies between the ISM practices was not evaluated.

Regarding the plausibility of the MOSAIC analysis activities, there are also some threats. First, we partially used the MOSAIC Toolbox to get the analysis activities results. Therefore, the evaluation of their plausibility is restricted. Furthermore, the plausibility from the software projects perspective could not be entirely evaluated. The benefit of the selected ISM practices cannot not be immediately identified in software projects, but have to be monitored over a certain period of time. Therefore, we cannot say that the support degree identified by us will support the software project to manage its critical situations. Some punctual experiences indicated this fact, but a systematic analysis of the benefit of the selected ISM practices could not be performed due to restricted time of the field study. Therefore, such an analysis is needed to verify if the support degree of an ISM practice for a SFM situationalFactor identified by us turns out to be correct for a software project.

8.4.4 Conclusion and Summary

We performed two case studies to evaluate the plausibility of the MOSAIC meta-models and of the modeling and analysis activities and a field study to evaluate the usefulness of MOSAIC.

The modeling of a high number of MOSAIC elements within the first case study led to an integration of the afore-mentioned PRs between each other and the integration of these PRs with the software project context. These integrations allowed us to automatically perform the MOSAIC analysis activities. Based on the created models, we automatically selected ISM practices, identified similar ISM practices or dependencies between ISM practices. Consequently, an integration according to the MOSAIC meta-models and modeling activities was possible and allowed the automation of the analysis activities.

Furthermore, the mapping of the CMMI-DEV practices to SFM situationalFactors in the second case study indicated that the selection of ISM practices was possible based on the relations between ISM practices and SFM situationalFactors. These relations were defined based on the mapping results obtained in the case study.

Finally, the implementation of several usage activities where MOSAIC could be applied within the software process improvement program of one organization indicated the usefulness of MOSAIC. We also partially evaluated the plausibility of the analysis activities.

8.5 MOSAIC Toolbox Evaluation

During the implementation of the MOSAIC Toolbox, we focused on the fulfillment of the defined functional requirements and less on non-functional requirements. However, besides the evaluation of the achievement of the functional requirements, we aimed to give an impression about the fulfillment of some non-functional requirements, such as the efficiency or usability. Therefore, we performed some activities to roughly evaluate these.

Based on a quality model [ISO/IEC 25010:2011 2011], we evaluated the quality in use of the MOSAIC Toolbox. Consequently, we evaluated the effectiveness, efficiency, satisfaction and usability attributes according to the following ordinal scale: “Low”, „Medium“ and „High“. The quality attribute “safety”, defined by the standard, is not relevant because there is no risk that the MOSAIC Toolbox could harm people.

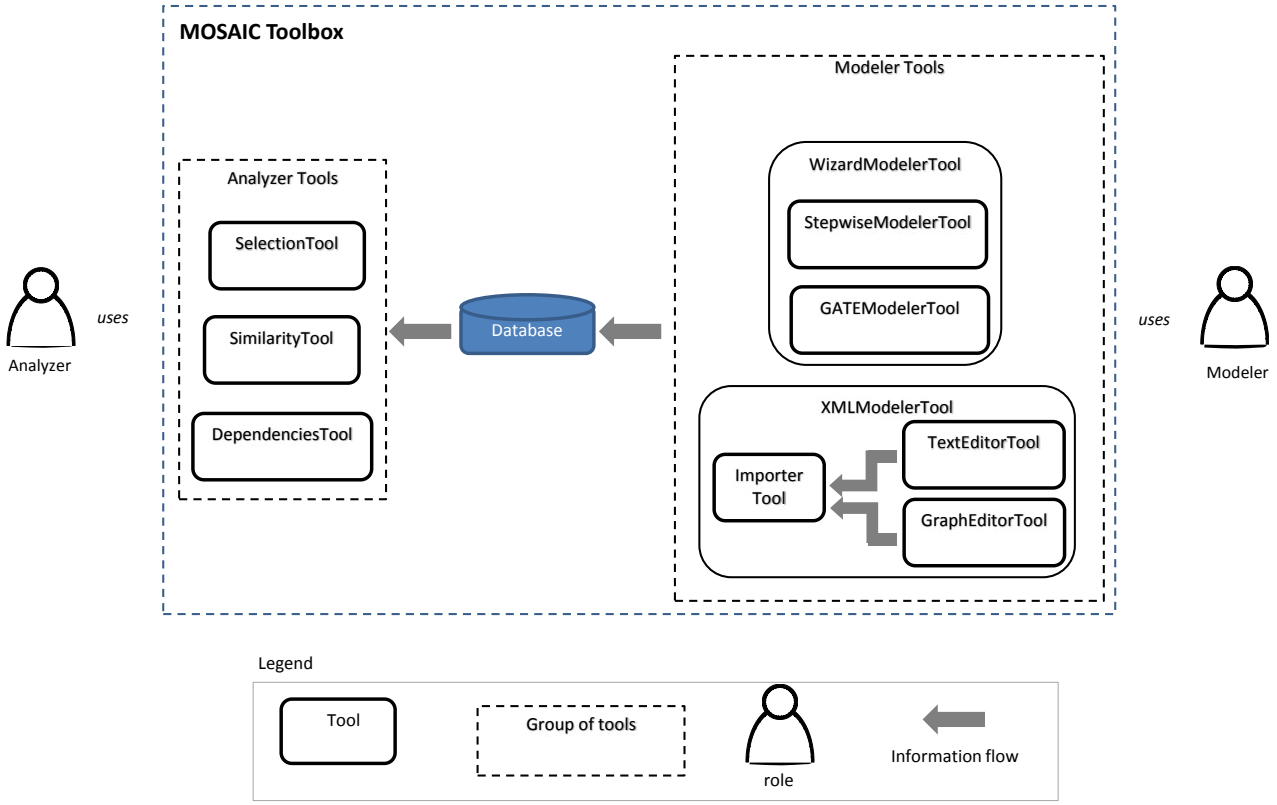


Fig. 57: Reminder – Overview MOSAIC Toolbox

8.5.1 Effectiveness

The effectiveness refers to the capability of a software application to allow users to achieve the specified requirements with accuracy and completeness.

To evaluate the effectiveness of the analyzer and modeler tools, we performed various tests defined according to the black-box testing method. For each functional requirement, we defined test cases to evaluate its accuracy. Based on the test results, we evaluated the effectiveness using a 3-point ordinal scale:

- **"High"**: More than 70% test cases are performed successfully.
- **"Medium"**: Between 30% and 70% test cases are performed successfully.
- **"Low"**: Less than 30% test cases are performed successfully.

8.5.1.1 Analyzer Tools

We systematically verified the results of the analyzer tools by defining two types of equivalence class partitioning tests:

- *Output based equivalence class partitioning tests*. To obtain different values for the support, similarity and dependency degree, we defined test cases that use as input CMMI-DEV, SPICE, ITIL, CMMI-SVC or COBIT practices and SFM situationalFactors from the situational factors framework [Clarke and O'Connor 2012]:
  - Over 1000 test cases evaluated the support degree of CMMI-DEV practices and eight SFM situationalFactors.

  - Over 180 test cases evaluated the similarity degree between ISM activityUnits of CMMI-DEV, SPICE, CMMI-SVC, ITIL and COBIT practices.
  - Over 50 test cases evaluated the dependency degree between CMMI-DEV practices.
- *Input based equivalence class partitioning tests*. We defined over 200 test cases based on dummy ISM practices to evaluate the results of all metrics. As the metrics are defined on different levels, we considered different **numbers** and **types** of elements on each level:
  - *ISM practice level*: We varied the **number** of ISM practices and define test cases that received as input two or more ISM practices. For example, we used these test cases to verify the computation of the similarity degree between two or more ISM practices.
  - *ISM activityUnit level*: We varied the **number** of ISM activityUnits within the ISM practices and defined test cases that received as input ISM practices with one or more ISM activityUnits. For example, we used these test cases to verify the computation of the similarity degree of these ISM practices based on the similarity of their ISM activityUnits.
  - *ISM practiceConcepts level*:
    - We varied the **number** of the ISM practiceConcepts within the ISM practices and defined test cases that received as input ISM practices with one or more ISM outputs. For example, we used these test cases to verify the computation of the output states of these ISM practices based on the state of their ISM outputs.
    - We varied the **type** of the ISM practiceConcepts and defined test cases that received as input ISM practices containing only ISM outputs, ISM roles or ISM inputs. For example, we used these test cases to verify the computation of the coverage degree between these ISM practices based on coverage degree of their ISM practiceConcepts.
  - *ICM concepts level*:
    - We varied the **number** of ICM concepts related to one ISM practiceConcept and defined test cases that received as input ISM practices with ISM practiceConcepts related to one or more ICM concepts. For example, we used these test cases to verify the dependencies between ISM practices based on the relations between these ICM concepts and their related ISM practiceConcepts.
    - We varied the **type** of similarity relations between ICM concepts and defined test cases that received as input ISM practices with ISM practiceConcepts related to ICM concepts related only by ICM composedOf, only by ICM generalizationOf or by both of them. For example, we used these test cases to verify the similarity degree between these ISM practices based on different similarity relations.

For the afore-mentioned tests, we defined the expected results. The test cases are 100% successfully performed, i.e., the obtained results were equal to the ones expected. Therefore, the effectiveness of the analyzer tools was evaluated as „High“.

8.5.1.2 Modeler Tools

The modeler tools support a Modeler to manually and semi-automatically perform the modeling activities. During the evaluation, we differentiated between the evaluation of the manual and semi-automated extraction of the MOSAIC elements.

Firstly, we evaluated the manual extraction of the MOSAIC elements with the `StepwiseModelerTool` except for the `GATEModelerTool`. As the MOSAIC elements have to be entered step by step, the probability of error is very small. However, we tested for each type of MOSAIC element,

that it was correctly saved in the database after its creation. Therefore, the effectiveness was evaluated as „High“.

Secondly, we evaluated the semi-automated extraction of the MOSAIC elements with the `XMLImporterTool`. The extraction is semi-automated as the data has to be modeled before it is automatically imported. Within a case study, we extracted over 2000 MOSAIC elements and we mainly used this tool for this purpose (section 8.4.1). After each automated import, we verified that the input is properly modeled in the database. Therefore, we evaluated the effectiveness as „High“.

Finally, we evaluated the semi-automated extraction of the ISM practice elements with the `GATEModelerTool` (also described in [Jeners et al. 2013b]). The extraction is semi-automated as the automated extracted ISM practice elements have to be corrected if necessary. We defined test cases by randomly selecting ISM processes with all their ISM practices. We aimed to obtain different results, i.e. different types and number of extracted normalized ISM practice elements. We totally selected 110 ISM practices of CMMI-DEV, SPICE, COBIT and IEC 61508.

Together with two students within our research department, we defined the expected results for each test case. Consequently, for each ISM practice we defined the expected number of ISM practice elements with their expected type (ISM activity, output, input, role and purpose). We performed the afore-mentioned test cases, compared the test results with the expected ones and counted the successful tests cases. 63% of the test cases were successfully performed and thus, the effectiveness was evaluated as “Medium”.

In the pattern recognition and information retrieval theory two special metrics are used to calculate the accuracy of an automated extraction of elements from the given text: precision[17] and recall[18]. Therefore, we also calculated these metrics to give a more precise information about the effectiveness of the `GATEModelerTool`.

To measure to what extent the extracted ISM practice elements were correctly extracted, we did not count the fault results (0 and 1 for an unsuccessful and successful test case). Instead, we used the cosine distance method to calculate the deviation between the extracted ISM practice elements and the correct ones. For example, if the element “establish a work breakdown structure” is a correct ISM activity and the `GATEModelerTool` extracted the element “establishes a work breakdown structure”, then the test case was not successful. Instead, the cosine distance determined a very small deviation.

The cosine distance method for two vectors of size n is defined as follows:

$$\boldsymbol{cos}_{\boldsymbol{p,q}} = \frac{\sum_{i=1}^{n} \boldsymbol{p_i} * \boldsymbol{q_i}}{\sum_{i=1}^{n} \boldsymbol{p_i^2} * \sum_{i=1}^{n} \boldsymbol{q_i^2}}$$

To calculate the cosine distance between the extracted and correct ISM practice element, we automatically created the corresponding vectors based on the stems[19] of the words. Therefore, we obtained a rational value between 0 and 1 and used it to compute the precision and recall metrics.

Table 50 lists the metrics’ results for the automated extraction of the ISM practice elements. The results are promising and indicates that the automated extraction is to some extent robust.

There are three main reasons why the automated extraction delivered fault results:

[17] Precision = fraction of retrieved instances that are correct

[18] Recall = fraction of correct instances that are retrieved

[19] Stem = the base form of a given word. Common examples of stems are: "writ" for written, "do" for doing, "categor" for categorize

- *Semantic of the elements*: The semantic is sometimes very important, especially for the identification of the ISM inputs and purposes. For example, the `GATEModelerTool` could not always differentiate between an ISM input and an output.
- *Complex language*: The language of the PRs is sometimes too complex for an automated extraction. For example, some PRs contain large sentences with more than 25 words.
- *Non-effective tools*: The applied GATE plug-ins did not always deliver the correct results. We observed that the ISM practice elements of ISM practice that contain verbs in the present continuous tense were not properly identified.

Metric	ISM practice element type				
	Activity	Output	Input	Role	Purpose
PRECISION	89%	91%	85%	85%	96%
RECALL	84%	82%	65%	85%	60%

Table 50: Extraction of ISM practice elements – Precision and recall metrics

8.5.2 Efficiency

The productivity of a software application depends on the amount of resources the users are spending in achieving their goals. The productivity is „High“ if the amount of spent resources of the users is small. Therefore, we analyzed the effort users need to perform the modeling and analysis activities according to the 3-point ordinal scale:

- **“High”:** A Modeler or Analyzer automatically performs an activity. Therefore, their effort is small.
- **“Medium”**: A Modeler or Analyzer semi-automatically performs an activity and their time effort is medium. The MOSAIC Toolbox automatically performs this activity, but the Modeler and Analyzer have to prepare or correct the results before or after the automation respectively. This preparation or correction is not time-consuming and thus, their effort to perform the activity is medium.
- **“Low”:** A Modeler or Analyzer semi-automatically performs an activity and their time effort is high. The MOSAIC Toolbox automatically performs this activity, but the Modeler and Analyzer have to prepare or correct the results before or after the automation respectively. This preparation or correction is time-consuming and thus, their effort to perform the activity is high.

The analyzer tools support an Analyzer to automatically perform the analysis activities. Consequently, we evaluated the productivity as „High“.

The modeler tools support a Modeler to manually and semi-automatically perform the modeling activities. We evaluated the productivity of each of these tools based on our experiences. For this evaluation, we assumed that the given input did not contain any fault data:

- The `StepwiseModelerTool` supported us step by step to manually perform all modeling activities. We had to perform several steps to create, retrieve, update or delete the MOSAIC elements. For example, for the creation of an ISM practice, we had to select the corresponding ISM process, then click on the “Create”-button, give the ISM practice name and the ISM practice short-name in a separate window and finally click the “Save”-button). Therefore, we estimated the productivity as “Low”.

- The `GATEModelerTool` supported us to semi-automatically extract the ISM practice elements. As its results were not highly accurate and if necessary needed to be corrected, we evaluated the productivity as "Medium".
- The `TextEditorTool` supported us to semi-automatically perform all modeling activities with the exception of the extraction of ICM concepts. We could use the functionality of a simple text editor (e.g. with copy-paste functions the Modeler can easily add and edit elements) to prepare the data to be automatically imported in the system. Therefore, we evaluated its productivity as "Medium".
- The `GraphEditorTool` supported us to extract the ICM concepts. The tool offers different support to search for the ICM concepts, to edit them by copying and pasting them. However, we differentiate between two cases:
    - *Small number of ICM concepts (under 500 ICM concepts)*. The tool was easy to use. We could create the ICM concepts and the similarity relations by drawing the corresponding elements on the canvas. Furthermore, the drawing canvas offers an overview of the existing ICM concepts and thus, supports the identification of ICM similar concepts. We evaluated the productivity as "Medium".
    - *High number of ICM concepts (over 500 ICM concepts)*. First, the editing of the ICM with a high number of ICM concepts became complex. This is because all the ICM concepts were not fitting the drawing canvas so that it became difficult to edit them. For example, it was very difficult to define an ICM composedOf between ICM concepts located in different parts of the canvas. Furthermore, the time response of the tool decreased. Therefore, we evaluated the productivity as "Low".

8.5.3 Satisfaction and Usability

The satisfaction of a software application refers to its capability to support users in a specific context of use. The usability refers to the capability of the software application to support users to achieve its goals with effectiveness, efficiency and satisfaction.

We organized a workshop in collaboration with the organization B (section 8.2). We aimed to evaluate the organization's satisfaction regarding the implemented functionality and usability of the MOSAIC Toolbox (presented also in the [Jeners and Lichter 2013]).

We conducted a workshop with five software process engineers of this organization and performed the following steps:

- *Presentation of MOSAIC*. We described the modeling activities and the resulted MOSAIC models, the identification of similar ISM practices and of dependencies between ISM practices. Not all analysis activities were presented because only the mentioned activities were implemented at the time of the workshop.
- *Usage of MOSAIC by the workshop participants*. The workshop participants identified similarities and dependencies between CMMI-DEV, ITIL and CMMI-SVC practices of interest.
- *Evaluation of the satisfaction and usability of the MOSAIC Toolbox*. To evaluate the satisfaction, we asked the participants to evaluate how relevant the analysis activities are for organizations working with multiple PRs. Then, we asked them to evaluate the usability of the MOSAIC Toolbox.

The participants evaluated the satisfaction according to a 3-point ordinal scale:

- **"High"**: The analysis activity is relevant for organizations working with multiple PRs.

- **"Medium"**: The analysis activity is nice to have for organizations working with multiple PRs.
- **"Low"**: The analysis activity is not needed for organizations working with multiple PRs.

The evaluation results showed that the participants acknowledged that organizations might have a „High" interest in the implemented analysis activities (Table 51). They showed a „High" interest for the identification of similar ISM practices especially on the computation of the highest coverage degree. At the time of the workshop, the organization aimed to evaluate the process compliance of the internal process PRs according to CMMI-SVC and to get certified. Therefore, the software process engineers were especially interested in the computation of the highest coverage of an internal process PR practice considering a set of CMMI-SVC practices. Finally, the identification of the output states of the ISM practices and the identification of dependencies between the ISM practices were also evaluated as „High".

We calculated the frequencies of the experts' evaluations and marked the highest value. Based on the average of these frequencies, we calculated the total frequency and also marked the highest value to evaluate the satisfaction. To summarize, the participants evaluated the satisfaction as „High".

Tools	Metrics	Satisfaction								
		Experts			Frequencies			Total Frequencies		
		High	Medium	Low	High	Medium	Low	High	Medium	Low
SimilarityTool	Similarity Degree	4	1	0	80%	20%	0%	77%	23%	0%
	Highest Practice Coverage	5	0	0	100%	0%	0%			
	Best Coverage	3	2	0	60%	40%	0%			
	Output State	4	1	0	80%	20%	0%			
DependencyTool	Dependencies	4	2	0	67%	33%	0%			

Table 51: Evaluation of the satisfaction of the MOSAIC Toolbox

Furthermore, the participants evaluated the usability of the MOSAIC Toolbox according to a 3-point ordinal scale:

- **"High"**: The MOSAIC Toolbox tool is easy to use.
- **"Medium"**: The MOSAIC Toolbox tool is moderate in use.
- **"Low"**: The MOSAIC Toolbox tool is difficult to use.

Tools	Metrics	Usability of the MosAIC Toolbox								
		Experts			Frequencies			Total Frequencies		
		High	Medium	Low	High	Medium	Low	High	Medium	Low
SimilarityTool	Similarity Degree	3	2	0	60%	40%	0%	45%	55%	0%
	Highest Practice Coverage		5	0	0%	100%	0%			
	Best Coverage	2	3	0	40%	60%	0%			
	Output State	4	1	0	80%	20%	0%			
DependencyTool	Dependencies	3	2	0	60%	40%	0%	60%	40%	0%
WizardModelerTool		1	0	4	20%	0%	80%	40%	20%	40%
XMLModelerTool		3	2	0	60%	40%	0%			

Table 52: Evaluation of the usability of the MOSAIC Toolbox

The usability of the `SimilarityTool` was evaluated as „Medium" and the usability of the `DependencyTool` as „High" (Table 52). To improve the usability, we collected the feedback of the five

experts and performed some improvements of the MOSAIC Toolbox. For example, we improved the visualization of the similarity results by sorting these results according to their similarities or by the visualization of an overview with the maximum similarity degree. The usability of the `StepwiseModelerTool` was evaluated as „Low“ and of the `XMLModelerTool` as „High“.

8.5.4 Summary and Future Work

Table 53 summarizes the evaluation of the quality attributes defined by the ISO/IEC 9126.

Quality Attributes	Tools			Evaluation		
				Low	Medium	High
Effectiveness	AnalyzerTool					✓
	ModelerTool		All tools except GATEModelerTool			✓
			GATEModelerTool		✓	
Productivity	AnalyzerTool					✓
	ModelerTool	WizardModelerTool	StepwiseModelerTool	✓		
			GATEModelerTool		✓	
		XMLModelerTool	TextEditorTool		✓	
			GraphEditor Tool (small no. of ICM concepts)		✓	
			GraphEditorTool (high no. of ICM concepts)	✓		
Satisfaction	AnalyzerTool					✓
Usability of the MOSAIC Toolbox	ModelerTool	WizardModelerTool		✓		
		XMLModelerTool				✓
	AnalyzerTool				✓	

Table 53: Summary – MOSAIC Toolbox Evaluation

To increase the quality of the MOSAIC Toolbox, the following aspects should be addressed in the future work:

- *Efficiency*. As the modeling of a high number ICM concepts becomes difficult with the `GraphEditorTool` within the `XMLModelerTool`, the usability of the `WizardModelerTool` have to be increased to efficiently support the Modeler in this modeling activity.
- *Satisfaction – Relevance of the analysis activities*. The participants asked for the implementation of an access management to protect information from unauthorized users. There exist commercial PRs that can be integrated in MOSAIC. Consequently, this should be available only for the organizations that have a license for it. Moreover, the internal process PRs have to be protected. Consequently, user accounts need to be implemented to protect the information from unauthorized users.
- *Usability of the MOSAIC Toolbox*. Five subjects evaluated the usability during the workshop. First, the number of the subjects was small. Second, they used the MOSAIC Toolbox only during the workshop. The MOSAIC Toolbox have to be used during a longer period of time to evaluate its usability. However, the MOSAIC Toolbox is only a proof-of-concept and its usability is only to some extent important. To achieve a production quality, various users (Modeler, Analyzer) and developers need to be involved to evaluate and improve its usability.

8.6 Summary

We performed several experiments, a case and a field study with the support of different subjects. Furthermore, we evaluated the MOSAIC Toolbox according to the quality in use model [ISO/IEC 25010:2011 2011].

The involvement of various experts in the evaluation activities was important to acquire a broader feedback and to use this expertise to improve MOSAIC. First, the involvement of the software process consultants was very important for our evaluation because of their high expertise. Furthermore, the involvement of academic researchers was also very important. They did not only evaluate MOSAIC based on their experiences with the definition and adoption of multiple PRs, but also from an academic perspective. Finally, the involvement of our industry partners helped us to evaluate MOSAIC usefulness, to exemplify its application and finally to evaluate the satisfaction and usability of the MOSAIC Toolbox.

Some students supported us in the development of the MOSAIC analysis activities too. However, we did not involve any students in the evaluation of the analysis activities results as we observed that a certain working experience in the adoption of the considered PRs is necessary.

Table 54 provides an overview of the evaluation performed with the experts mentioned in the section 8.2. We mark the PRs that are used in the evaluation activities. Therefore, we demonstrated the flexibility of MOSAIC, i.e. that the integration of PRs for different software areas is possible. Based on the integration of all these PRs, we could evaluate the plausibility and usefulness of MOSAIC. For each MOSAIC analysis or modeling activity, we document if it is evaluated within an experiment, case or field studies, if its plausibility, usefulness and flexibility was entirely (“yes”), partially (”partially”) or not (“no”) verified. Finally, the last column illustrates that all the analysis and modeling activities were considered in the evaluation of the MOSAIC Toolbox.

MOSAIC Activities		Considered PRs					Evaluation activities		Evaluation goals			
		CMMI-DEV	SPICE	CMMI-SVC	ITIL	COBIT	Experiments	Case / Field studies	Plausibility	Usefulness (Field study)	Flexibility (Integration of PRs)	MOSAIC Toolbox
							Operation	Operation				
Selection of ISM practices based on SFM situationalFactors	Support Metrics	✓	-	-	-	-	-	✓	partially	yes	no	yes
Identification of similar ISM practices	Similarity Metrics	✓	✓	✓	✓	✓	✓	✓	yes	yes	yes	yes
	Coverage Metrics	✓	-	-	-	-	-	✓	partially	yes	yes	yes
	Output State Metrics	-	-	-	-	-	-	-	no	no	no	yes
Identification of dependencies between the ISM practices	Dependency Metrics	✓	-	-	-	-	✓	-	yes	no	no	yes
Modeling of ISMs, ICM and SFM		✓	✓	✓	✓	✓	-	✓	partially	no	yes	yes

Table 54: Overview of the MOSAIC activities and the corresponding evaluation activities

There are several threats to validity to be mentioned for each MOSAIC analysis or modeling activity:

- Selection of ISM practices based on SFM situationalFactors:
  - The plausibility was not evaluated during controlled experiments, but only within a case and field study and thus, the controllability and the plausibility is restricted.
  - The plausibility evaluation within the case study is restricted as the case study experiences and results were used for the development of this analysis activity and of the modeling activities.

    - The plausibility evaluation within the field study to verify if the recommended ISM practices bring the promised benefit in the projects could not be performed.
- Identification of similar ISM practices:
    - The plausibility of the identification of similar ISM practices based on similarity metrics was not evaluated for the entire MOSAIC models. As this analysis activity highly depends on the MOSAIC models, the identification of similar ISM practices for other parts of PRs that were not considered in the experiments could lead to wrong results.
    - The plausibility of the identification of similar ISM practices based on the coverage metrics was not evaluated during controlled experiments. It was partially evaluated in the field study.
    - The plausibility and the usefulness of the identification of similar ISM practices based on the output state metrics was not evaluated in experiments, case or field studies. However, its usefulness was evaluated in a workshop with one organization (see MOSAIC Toolbox Evaluation).
- Identification of the dependencies between ISM practices:
    - The usefulness could not be evaluated in a field study, but it was evaluated in a workshop with one organization (see MOSAIC Toolbox Evaluation).
- Modeling of ISMs, ICM and SFM:
    - The plausibility of the ISMs and ICM was only evaluated by us and not by other experts. However, we evaluated the MOSAIC results during the experiments, case and field studies based on the built models.
    - The modeling activities were only performed by us.

Therefore, from a statistical point of view, the number of evaluation activities and of involved expertise is small. Further evaluation needs to be performed to improve MOSAIC:

- Further experts need to be involved to evaluate the plausibility of the analysis activities in controlled experiments.
- Analysis activities need to be broadly applied in the industry and their impact need to analyzed over the years.
- The MOSAIC models need to be systematically reviewed by experts with a deep knowledge of the PRs and software project context.
- The modeling activities need to be performed by experts to objectively evaluate them.
- Further PRs and SFM situationalFactors need to be integrated into MOSAIC and used in the future evaluation activities.

We close with a summary of the main results and lessons learned gained during these evaluation activities:

- The integration of multiple PRs with the software project context within the modeling activities is possible to allow the automation of the analysis activities.
- The integration of PRs for different software areas is possible.
- The analysis activities deliver promising results.
- The modeling activities require a high expertise of the software engineering.
- The modeling activities require a high effort as the PRs have different structure and terminology. This high effort is also caused by the modeling of PRs on a very detailed level.

- The usage of the MOSAIC Toolbox supports organizations to get consistent results. Furthermore, the more experts participate in the creation of the MOSAIC models, the better the models are and the better they can be used to deliver reliable results.

9 Conclusions

We close this work by a short description of the contributions that we achieved with MOSAIC. A summary of its main limitation and a short outlook is also given. Finally, we give an overview of MOSAIC and state the final conclusion.

9.1 Contributions

MOSAIC supports organizations that deal with one or more PRs to adopt and assess their software processes according to these PRs.

There are numerous approaches that support organizations in the selection of best suited PRs for their software process improvement program or in the simultaneous usage of these PRs. However, there is no single approach being universally applicable. Our tool based approach, called MOSAIC facilitates the selection and usage of PRs by an integration of the PRs with each other and an integration of the PRs with the software project context.

This integration is possible based on three meta-models and their instantiations: ISM (Integrated Structure Model), ICM (Integrated Concept Model) and SFM (Situational Factors Model). The central model is the ICM. This integrates the various PRs at a fine-grained level by extracting their concepts. These concepts are related to the ISMs which are used to normalize the structure of the PRs. Furthermore, the concepts are related to the SFM that contains situational factors to characterize the software project context. Due to this design, MOSAIC has various benefits, of course within the boundaries of its meta-models. According to the evaluation performed and our experiences with MOSAIC, we mention the following benefits:

- **Various analyses of PRs are enabled.** MOSAIC allows running different automations on its models depending on the interests of the organizations that work with multiple PRs. These are offered by the MOSAIC Toolbox and are an implementation of the following analysis activities: selection of practices based on situational factors that describe the software project context, identification of similar practices or of dependencies between them. As their name says, the *analysis activities* can help organizations to analyze various PRs and to make software process improvement decisions.
- **Further analyses of PRs are possible.** The implementations of the analysis activities are only some examples of automations that MOSAIC offers. Such an example is the computation of the similarity degree between two or more practices based on the similarity metrics. Other implementations are also possible based on MOSAIC meta-models, models and metrics. For example, an algorithm can be developed to identify a mapping between the processes of two PRs based on the similarity metrics. Furthermore, other analysis activities can be implemented by extracting the information from the MOSAIC models. For example, the identification of the value of a reference PR for an organization can be developed based on MOSAIC.
- **Multilateral comparison of ISM practices is supported.** The related approaches always consider a bilateral comparison between PRs, practices or practice elements. Not only a bilateral comparison, but also the comparison of three or more practices is supported by MOSAIC. The MOSAIC design with its ICM that connects all PRs concepts allows us to compare more than two practices at once by identifying their similar concepts. This comparison can support organizations to identify the commonalities of more than two PRs in a single operation.

- **MOSAIC can be extended and applied for further domains.** MOSAIC can be used to support activities that are not related to the usage of PRs. The MOSAIC models and metrics have to be extended, but can inspire the development of such approaches. For example, the similarity metrics can inspire the development of approaches to identify redundancies in the software requirements.
- **MOSAIC models are extendable.** New PRs or situational factors can be easily integrated into MOSAIC by defining their relations to the ICM. For example, agile PRs can be integrated into MOSAIC as this might address a larger part of the software development community. Moreover, factors that influence how IT supports the business could be integrated into MOSAIC. These could be then mapped to COBIT concepts that are specific to business and allow an automated selection of their corresponding practices. This could be relevant for many organizations as COBIT is a commonly used PR. The integration of new PRs becomes easier when the ICM contains a high number of concepts. This is because the ICM already contains many of the corresponding concepts and few new concepts have to be added. In this case, only the relations to the existing concepts have to be defined. Not only the design, but also the MOSAIC Toolbox, supports an easy extension of the MOSAIC models by modeling the new PRs in XML format and importing them in the system.
- **MOSAIC is not limited to software development.** We demonstrated that not only PRs for the software development, but also for other software areas can be integrated into MOSAIC. PRs, such as ITIL or COBIT are modelled and analyzed in MOSAIC during its evaluation. Furthermore, we argue that the situational factors extracted from an existing framework [Clarke and O'Connor 2012] are not only valid for the software development, but also for other software areas. Hence, the selection of practices based on these situational factors can be used in these areas.
- **MOSAIC maturity grows**. The more PRs are integrated into MOSAIC, the better is the quality of ICM and the easier is the integration of new PRs. Practices from multiple PRs provide the use context of a PR concept and thus, a Modeler can better understand its meaning and can better extract new concepts or improve the ICM. Therefore, the ICM acts as a dictionary, where the context for each concept is described in the PRs.
- **An ontology of PRs terms is created.** Multiple PRs are integrated based on an extraction and relation of their concepts. Therefore, the ICM defines an ontology of terms of the software areas. It unifies the knowledge of the multiple PRs and thus, can serve as a knowledge base for different automations.
- **MOSAIC is maintainable.** Modifications of the MOSAIC meta-models, models or of the analysis activities can be easily performed. When the PRs are changed and a new version of a PR exists, the changes to the models can be easily performed by modifying only the changed elements and their relations to the concepts. Furthermore, the MOSAIC Toolbox is also maintainable. First, the analysis activities can also be easily modified as there are dedicated components that handle the analysis activities. Furthermore, when the meta-models are changed, new code can be generated based on these meta-models and integrated in the MOSAIC Toolbox.

Another contribution of this work beyond MOSAIC, is the description of various **usage activities** that can be relevant for the software process improvement of organizations (chapter 7). Therefore, organizations receive a check list that reveals various cases they need to pay attention to when dealing with PRs. For each such usage activity, we described their benefits and how MOSAIC can support its application. Furthermore, we also gave guidelines how to apply them concretely. For example, we provided guidelines how to identify the value of PRs for an organization (section 7.3) or we describe various alternatives how to create a repository with multiple PRs (section 7.6.2).

Although various experts participated in the evaluation of MOSAIC and they acknowledged the contribution of MOSAIC for organizations that work with one or more PRs, there are some limitations that can be addressed in the future work.

From a statistical point of view, the number of evaluation activities and of involved experts is small. A broader involvement of experts from research and industry in the evaluation and improvement of MOSAIC analysis activities and in the reviews of the MOSAIC models is needed. Furthermore, the implemented analysis activities need to be broadly applied in the industry and their impact needs to be analyzed over the years.

Finally, the MOSAIC Toolbox offers a prototypical implementation of MOSAIC to demonstrate that the integration of PRs and software project context is possible to allow various automations. This tool support can be improved and extended in the future work. Further implementations of the analysis activities or further analysis activities can be implemented by the MOSAIC Toolbox to support organizations to work with multiple PRs. Furthermore, the MOSAIC Toolbox has to be improved to fulfil not only functional, but also non-functional requirements, such as usability or performance.

9.2 Summary

An approach, called MOSAIC, is proposed to integrate multiple PRs with the software project context (Fig. 58). This integration is based on a normalization of the structure and terminology of the various PRs. It is performed at a conceptual level by extracting concepts and relating them to practice elements, as well as to a wide variety of situational factors that describe software development settings, domains and contexts [Clarke and O'Connor 2012].

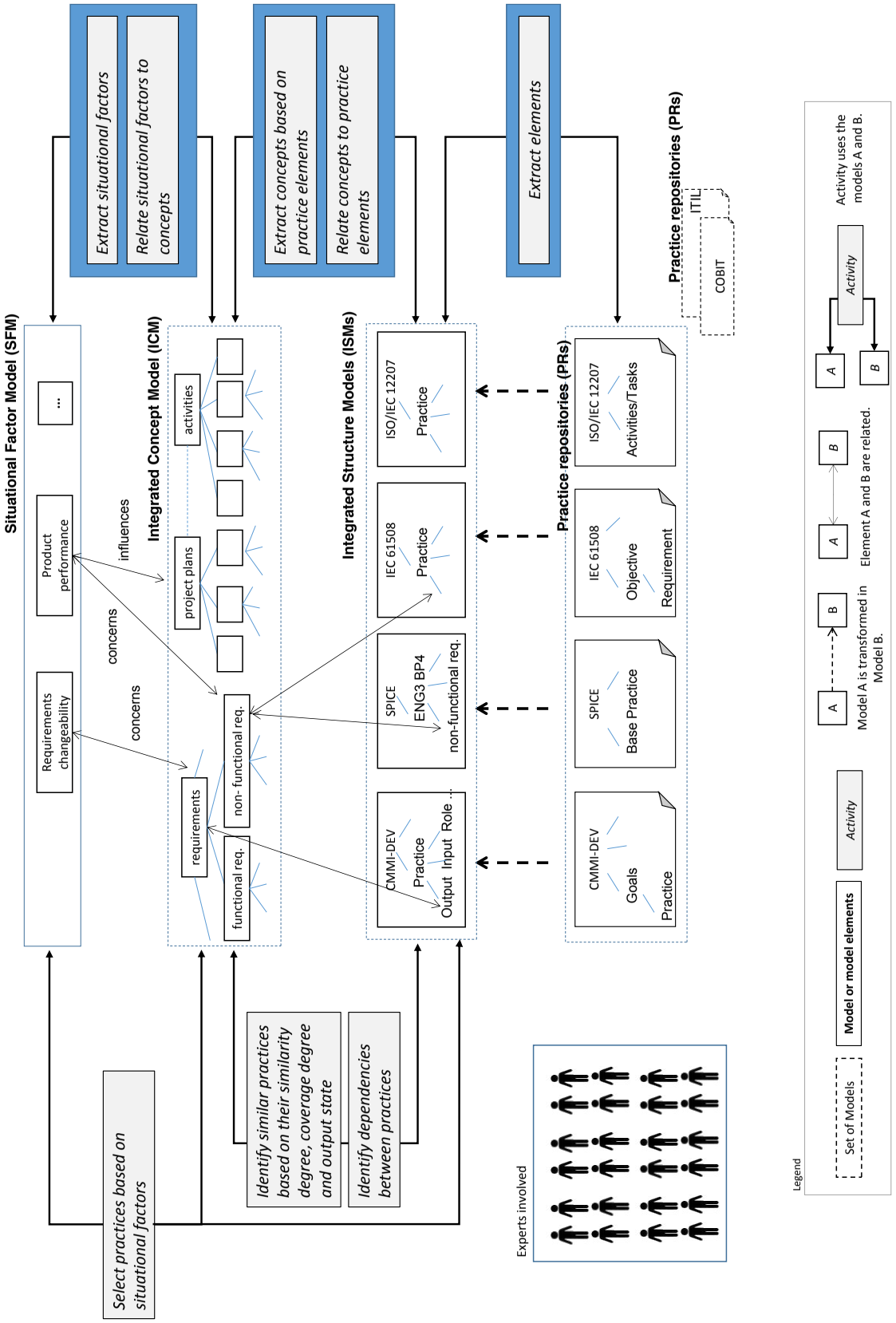


Fig. 58: MOSAIC summary

In addition, MOSAIC offers a wide variety of implementations of analysis activities, that can be automatically performed on a collection of diverse PRs. These activities can be utilized by organizations when attempting to make key process decision for an effective and efficient adoption of PRs and assessment based on PRs. An effective adoption and assessment is possible because only practices are selected that are best suited for software projects of an organization. This selection of practices is based on the situational factors that characterize the software project context. Furthermore, the identification of similar practices and dependencies between these practices support an efficient adoption and assessment. Redundancies in the adoption and assessment can be avoided by the computation of the similarity and coverage degree between practices. Helpful information for the adoption can be provided by the identification of similar practices concerning a certain output and its working state. Finally, inconsistencies and dependencies in the software processes can be well managed by the identification of the dependencies between the practices.

Various experts were engaged to evaluate the approach and its contribution. We totally involved 24 experts from different organizations, countries and with a different expertise: software consultants from consultancy organizations, academic researchers and the employees of two organizations.

In the absence of published guidance with respect to such automations, we presented a robust approach for relating software processes and software development contexts. Several publications and distinctions for this approach recognized its value and contribution within the research community. Furthermore, early evidence from industrial application suggested that the approach brings also benefits in practice. Therefore, MOSAIC can serve as a valuable approach for different software practitioners involved in the software process improvement initiatives of organizations.

10 Appendices

In this chapter, we give more details about the MOSAIC Toolbox. We describe the functional requirements and give guidelines how to use the MOSAIC Toolbox.

10.1 MOSAIC Toolbox requirements

The MOSAIC Toolbox requirements refer to functional requirements and are illustrated by the use cases in Fig. 39. In the following, we describe these requirements using an use case template, the most common used method to describe a use case [Hoffmann 2013]. The table oriented use case template contains different parameters that define the type of information to be described (Table 55). This information is categorized in stages:

- *First stage* contains general parameters, such as the name of the use case or the actors involved in it.
- *Second stage* contains parameters that describe the innards of a use case, such as the standard steps inside an use case. Furthermore, the conditions for the start and at the finish of the use case are also described.
- *Third stage* contains a description of the alternatives and exceptions from the standard flow.

Use Case Template	
First Stage	
ID	<ID>
Name	<Name of the Use Cases>
Actors	<Names des actors of the Use Case>
Goal	<Short description of the goal of the Use Case>
Short Description	<Short description of the Use Case>
Second stage	
Precondition	<List of pre conditions>
Postconditions	<List of post conditions>
Standard event flow	<Descriptions of the main steps that lead the actor to the goal.> 1. [Step 1] 2. [Step 2] …
Additions to the standard event flow	<Additional hints to better understand the standard flow of events. You may also add exceptions here. This should be simple text. You should refer to the steps in the standard flow.>
Third stage	
Alternatives and	<Please take every alternative and exception into account. Some of them may

Exceptions	lead to the goal, some may end in an error state.> 1a. [Condition 1] 1a1. [Step 1.1] 1a2. [Step 1.2] 1b. [Condition 2] 1b1. [Step 2.1] …

Table 55: Use case template [Hoffmann 2013]

In the following, we describe the use cases that define the interactions between an actor – Modeler or Analyzer – and the MOSAIC Toolbox.

10.1.1 Use Cases for a Modeler

As the following use case description refers to IS Meta-Model elements that are not often addressed in this work, we remind about their purpose.

ISM practice repository elements are the elements that group or refer to the ISM practices of the multiple PRs. Such elements are **ISM practiceRepositories**, **categories**, **processes** and **practices**.

Use-Case	
ID	I
Name	Extract ISM elements.
Actors	Modeler
Goal	Structure normalization of a PR
Short Description	The ISM elements related to one ISM practice are extracted and saved in the system.
Precondition	An ISM practice is identified by the Modeler.
Postcondition	All ISM elements related to the ISM practice are saved in the system.
Standard event flow	1. Modeler requires getting an overview of the ISM practiceRepositories that exist in the system. 2. System visualizes the requested elements. 3. Modeler extracts the corresponding ISM practice repository elements: 3.1. Modeler extracts the ISM practiceRepository and enters it in the system. 3.2. System saves and visualizes the ISM practiceRepository. 3.3. Modeler selects the ISM practiceRepository, extracts the corresponding ISM category and enters it in the system. 3.4. System saves and visualizes the ISM category. 3.5. Modeler selects the ISM category, extracts the corresponding ISM

	process and enters it in the system. 3.6. System saves and visualizes the ISM process. 3.7. Modeler selects the ISM process, extracts the ISM practice and enters it in the system. 3.8. System saves and visualizes the ISM practice. 4. Modeler selects the ISM practice and decides to manually extract the ISM practice elements from it: 4.1. Modeler enters an ISM activityUnit in the system. 4.2. System visualizes the ISM activityUnit. 4.3. Modeler selects an ISM activityUnit and extracts its ISM practice elements: 4.3.1. Modeler extracts an ISM activity and enters it in the system. 4.3.2. System saves and visualizes it. 4.3.3. Modeler extracts an ISM explicit output and enters it in the system. 4.3.4. System saves and visualizes it. 4.3.5. Modeler extracts an ISM explicit input and enters it in the system. 4.3.6. System saves and visualizes it. 4.3.7. If an ISM role exists, the Modeler extracts and enters it in the system. 4.3.8. System saves and visualizes it. 4.3.9. If an ISM purpose exists, the Modeler extracts and enters it in the system. 4.3.10. System saves and visualizes it. 4.3.11. Modeler continues with step 4.3.1 until no ISM roles, purposes and ISM explicit artifacts for the current ISM activityUnit need to be extracted anymore. 4.4. Modeler continues with step 4.3 until the ISM practice does not contains any ISM activityUnits anymore. 5. Modeler selects the ISM practice and manually extracts the ISM implicit artifacts from the PRs' description for this ISM practice if these exist: 5.1. Modeler selects an ISM activityUnit and extracts its ISM implicit artifacts: 5.1.1. Modeler extracts an ISM implicit output and enters it in the system. 5.1.2. System saves and visualizes it. 5.1.3. Modeler extracts an ISM implicit input and enters it in the system. 5.1.4. System saves and visualizes it. 5.1.5. Modeler continues with step 5.1.1 until no implicit ISM practice artifacts need to be extracted anymore.
Alternatives and Exceptions	**4.a. Condition: If the Modeler decides to semi-automatically extract the ISM practice elements based on the ISM practice:** a.1. UC Extract semi-automatically ISM practice elements.

Use-Case	
ID	I.a
Name	Extract semi-automatically ISM practice elements.
Actors	Modeler
Goal	Structure normalization of a PR
Short Description	For an ISM practice, its ISM activities, purposes, roles and explicit artifacts are automatically extracted, grouped in ISM activityUnits, corrected by the Modeler and saved in the system.
Precondition	The ISM practice already exists in the system.
Postcondition	The ISM practice elements are saved in the system.
Standard event flow	1. Modeler selects an ISM practice and decides to semi-automatically extract its ISM practice elements. 2. System extracts the ISM activityUnits, their ISM activity, purposes, roles and explicit artifacts, saves and visualizes them. 3. Modeler verifies the extracted ISM practice elements and if they are correctly extracted, he does not perform any steps anymore.
Alternatives and Exceptions	**3.a. Condition: If the extracted ISM practice elements are not correctly extracted:** 3.a.1: Modeler selects the corresponding ISM activityUnit and creates new ISM practice elements, updates or deletes the incorrect ones. 3.a.2: System saves and visualizes the results.

As the following use case descriptions addresses the different roles that an ICM concept can have in a generalizationOf-mono-hierarchy, we remind about their definition.

An **ICM abstract concept** is a ICM general concept and is the root of the generalizationOf-mono-hierarchy.

An **ICM synonym concept** for an ICM concept has the same semantic meaning as the ICM concept and refers to the same ICM concept.

An **ICM descendant concept** for an ICM concept conc is a ICM special concept on the many paths between the ICM concept conc and ICM concepts at lower levels in the generalizationOf-mono-hierarchy

Use-Case	
ID	II.1
Name	Extract ICM concepts based on ISM practiceConcepts.
Actors	Modeler

Goal	Terminology normalization of a PR.
Short Description	For an ISM practiceConcept, its ICM concepts are extracted and saved in the system.
Precondition	The ISM practiceConcept already exists in the system.
Postcondition	Extracted ICM concepts are saved the system.
Standard event flow	1. Modeler requires to get an overview of the ICM concepts that exist in the system. 2. System visualizes the requested elements. 3. Modeler identifies an ICM concept in the ISM practiceConcept. 4. Modeler searches the ICM concept in ICM: 4.1. Modeler enters the ICM concept name in the system. 4.2. System searches the ICM concept and notifies the Modeler that the ICM concept does not exists. 4.3. Modeler identifies its ICM abstract concept and searches for an ICM synonym concept in the generalizationOf-mono-hierarchy. 4.4. System visualizes the generalizationOf-mono-hierarchy. 5. Modeler does not find an ICM synonym concept and creates the ICM concept and its ICM abstract concept: 5.1. Modeler creates the ICM concept. 5.2. System saves and visualizes it. 5.3. Modeler creates its ICM conceptCategory if this does not exist. 5.4. System saves and visualizes it. 5.5. Modeler creates its ICM abstract concept if this does not exist. 5.6. System saves and visualizes it. 6. Modeler creates the relations of the ICM concept and its ICM abstract concept: 6.1. Modeler identifies ICM similar concepts for the ICM concept and defines the corresponding similarity relations (ICM generalizationOf and composedOf). 6.2. System saves the similarity relations and visualizes them. 6.3. If the ICM abstract concept was created, the Modeler relates it with SFM situationalFactors. 6.4. System saves the relations and visualizes them. 6.5. Modeler verifies if the relations between the SFM situationalFactor and the ICM abstract concept of the current ICM concept are also valid for this current ICM concept. If not, he relates all the ICM descendant concepts in the generalizationOf-mono hierarchy except the current ICM concept with the SFM situationalFactor. 6.6. System saves the relations and visualizes them. 6.7. If the ICM abstract concept was created, the Modeler identifies its ICM similar concepts and defines the its similarity relations. 6.8. System saves the similarity relations and visualizes them.
Alternatives and Exceptions	**4.2.a. Condition: If the ICM concept exists:** 4.2.a.1: Modeler continues with step 3 until no ICM concepts can be

	identified anymore. **5.a. Condition: If the ICM synonym concept exists:** 5.a.1: Modeler continues with step 3 until no ICM concepts can be identified anymore.

Use-Case	
ID	II.2
Name	Relate ISM practiceConcepts to ICM concepts.
Actors	Modeler
Goal	Structure normalization of a PR
Short Description	The extracted ICM concepts are related to an ISM practiceConcept.
Precondition	The ICM concepts that need to be related to an ISM practiceConcept already exist in the system. The ISM practiceConcept exists in the system as well.
Postcondition	The relations between the ICM concepts and the corresponding ISM practiceConcept are saved in the system.
Standard event flow	1. Modeler requires to get an overview of the ISM practiceConcepts that exist in the system. 2. System visualizes the requested elements. 3. Modeler selects the ISM practiceConcept. 4. System displays all ICM concepts already contained in the system. 5. Modeler searches for an ICM concept. 6. System finds the ICM concept and visualizes it. 7. Modeler relates the ICM concept to the ISM practiceConcept. 8. System visualizes the related elements. 9. Modeler continues with the step 5 until all relevant ICM concepts are related to the ISM practiceConcept.
Alternatives and Exceptions	-

Use-Case	
ID	III.1
Name	Extract SFM situationalFactors.
Actors	Modeler
Goal	Modeling of the software project context
Short Description	A SFM situationalFactor is extracted and saved in the system.

Precondition	The SFM situationalFactor is identified by the Modeler.
Postcondition	The SFM situationalFactor is saved in the system.
Standard event flow	1. Modeler requires to get an overview of all SFM situationalFactors that exist in the system. 2. System visualizes the requested elements. 3. Modeler identifies the SFM situationalFactor and enters it. 4. System saves and visualizes the SFM situationalFactor.
Alternatives and Exceptions	-

Use-Case	
ID	III.2
Name	Relate SFM situationalFactors to ICM concepts.
Actors	Modeler
Goal	Relation of PRs to software project context
Short Description	A SFM situationalFactor is related to ICM concepts.
Precondition	The SFM situationalFactor and the ICM concepts that can be related to it exist in the system.
Postcondition	The relations between the SFM situationalFactor and the ICM concepts are saved in the system.
Standard event flow	1. Modeler requires to get an overview of the SFM situationalFactors and of the ICM concepts that exist in the system. 2. System visualizes the requested elements. 3. Modeler selects a SFM situationalFactor and searches for a corresponding ICM abstract concept. 4. System visualizes the found ICM abstract concept. 5. Modeler identifies the reasoning for a relation between the ICM abstract concept and the SFM situationalFactor and observes that this relation is valid for all ICM descendant concepts of the ICM abstract concept. 6. Modeler relates the ICM abstract concept to the SFM situationalFactor. 7. System visualizes the related elements.
Alternatives and Exceptions	**5a. Condition: If the relation between the SFM situationalFactor and the ICM abstract concept is not valid for all its ICM descendant concepts:** 5a.1: Modeler searches for ICM descendant concepts for which the relation is valid. 5a.2: System visualizes these ICM descendant concepts. 5a.3: Modeler relates these ICM descendant concepts with the SFM situationalFactor.

	5a.4: System visualizes the related elements.

10.1.2 Use Cases for the Analyzer

Use-Case	
ID	1
Name	Select ISM practices based on SFM situationalFactors.
Actors	Analyzer
Goal	Selection of best suited ISM practices for a software project based on its context.
Short Description	ISM practices are selected for a software project context characterized by a SFM situationalFactor.
Precondition	The SFM situationalFactor and the ISMs of the PRs already exist in the system.
Postcondition	ISM practices from multiple PRs together with their support degree for a SFM situationalFactor are identified and visualized.
Standard event flow	1. Analyzer requires to the get an overview of the SFM situationalFactors and ISM practiceRepositories that exist in the system. 2. System visualizes the requested elements. 3. Analyzer selects a SFM situationalFactor and the ISM practiceRepositories to be considered for the selection. 4. System computes the support degrees of the ISM practices of each selected ISM practiceRepository and visualizes all the ISM practices where the support degree is “Strong” or “Medium”.
Alternatives and Exceptions	-

Use-Case	
ID	2.1
Name	Identify the similarity degree of ISM practices.
Actors	Analyzer
Goal	Identification of similar ISM practices
Short Description	The similarities between two or more ISM practices are identified.
Precondition	The ISM practices from multiple PRs of interest and their related ISM elements are contained in the system.
Postcondition	The similarity degrees on the different levels (ISM practices, activityUnits, practiceConcepts and ICM concepts) are visualized. Furthermore, their common ICM abstract concepts are visualized.

Standard event flow	1. Analyzer requires to get an overview of the ISM practices that exist in the system. 2. System visualizes the requested elements. 3. Analyzer selects two or more ISM practices and chooses the bilateral comparison. 4. System computes the similarity degrees of each two ISM practices from different PRs, identifies their ICM abstract concepts and visualizes the results.
Alternatives and Exceptions	**3a. Condition: If the Analyzer does not choose the bilateral comparison:** 3a.1: System computes the similarity degree of all selected ISM practices at once and visualizes the results.

In the following use case description, the Analyzer aims to compute the highest and best coverage. Therefore, we remind about what these mean.

The **highest coverage** refers to a practice that has the maximum coverage degree in a considered set of practices.

The **best coverage** refers to the minimum subset of practices with a coverage degree of 1 in a considered set of practices.

Use-Case	
ID	2.2
Name	Identify the coverage degree of two sets of ISM practices.
Actors	Analyzer
Goal	Identification of similar ISM practices
Short Description	The coverage degree between ISM practices, the highest coverage or the set of ISM practices with the best coverage in a set of ISM practices are identified. Furthermore, the similarities and differences for the highest and best coverage are also identified.
Precondition	The ISM practices and their related ISM elements are contained in the system.
Postcondition	The ISM practices and their coverage degrees on the different levels (ISM practices, practiceConcepts and ICM concepts) are visualized. Furthermore, common, different and ICM similar concepts are visualized.
Standard event flow	1. Analyzer requires to get an overview of the ISM practices that exist in the system. 2. System visualizes the requested elements. 3. Analyzer selects two or more ISM practices from different ISM practiceRepositories. 4. Analyzer decides to compute the coverage degree. 5. System identifies the coverage degrees of two sets of ISM practices

	from different PRs. As the coverage metrics are not symmetric, each set of ISM practices from a PR is considered as reference and forms the first set. The second set contains all other selected ISM practices. The system visualizes the results.
Alternatives and Exceptions	**4a. Condition: If the Analyzer decides to compute the highest coverage degree:** 4a.1: System identifies the ISM practice with the highest coverage degree in the set of all selected ISM practices. It also identifies the common, different and ICM similar concepts and visualizes them. **4b. Condition: If the Analyzer decides to compute the best coverage degree:** 4b.1: System identifies the set of ISM practices with the best coverage degree in the set of all selected ISM practices. It also identifies the common, different and ICM similar concepts and visualizes them.

Use-Case	
ID	2.3
Name	Identify the output state of ISM practices.
Actors	Analyzer
Goal	Identification of similar ISM practices
Short Description	Similarities between ISM practices considering an ISM output are identified.
Precondition	The ISM practices and their related ISM elements are contained in the system.
Postcondition	ISM practices from multiple PRs together with their output states are identified.
Standard event flow	1. Analyzer requires to get an overview with the ICM concepts that exist in the system. 2. System visualizes the requested elements. 3. Analyzer selects an ICM concept. 4. System computes the output states of the ISM practices that contain ISM outputs related to this ICM concept or to ICM similar concepts and visualizes the results.
Alternatives and Exceptions	-

Use-Case	
ID	3
Name	Identify the dependencies between ISM practices.

Actors	Analyzer
Goal	Identification of dependencies between ISM practices
Short Description	Dependencies between the ISM practices considering certain PRs are identified.
Precondition	The ISMs of PRs of interest are contained in the system.
Postcondition	Dependencies of the selected ISM practices considering the selected PRs are visualized.
Standard event flow	1. Analyzer requires to get an overview with the ISM practiceRepositories that exist in the system. 2. System visualizes the requested elements. 3. Analyzer selects the ISM practiceRepositories to be considered and the ISM processes for which the dependencies of their ISM practices need to be identified. 4. Analyzer decides to identify the dependencies with the dependency degree "Strong". 5. System identifies the "Strong" dependencies of each ISM practices of the selected ISM processes and visualizes them.
Alternatives and Exceptions	**4a. Condition: If the Analyzer decides to identify dependencies with the dependency degree "Medium":** 4a.1: System identifies the "Medium" dependencies of each ISM practices of the selected ISM processes and visualizes them.

10.2 MOSAIC Toolbox Handbook

In the following, we give detailed guidelines how to use the tools within the MOSAIC Toolbox.

10.2.1 Running Example

After a description of the MOSAIC Toolbox tools, we illustrate them by using as input the following ISM practices within our scenario related to the software quality:

- CMMI-DEV PPQA SP1.2.5 "Identify each case of noncompliance found during evaluations."
- CMMI-DEV PPQA SP2.1 "Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers."
- SPICE SUP.1 BP9 "Ensure resolution on non-conformances."

10.2.2 Modeler Tools

A Modeler uses the modeler tools to save the ISM, ICM and SFM elements in the database. For this purpose, he can be guided by the `WizardModelerTool` and save the data directly in the database or can use the `XMLModelerTool` to model the data in XML format and then import it into the database.

10.2.2.1 WizardModelerTool

The Modeler extracts the ISM elements by using the `StepwiseModelerTool` and the `GATEModelerTool`. He enters each ISM element step by step (Create). For the ISM practice elements, he can parse the ISM practice and get them at once (Parse) (Fig. 59). Although the results of the automated extraction are useful to some extent, the Analyzer has to validate or correct the extracted ISM practice elements. The tree elements are all editable, i.e. their type (ISM input, output or role) can be easily changed by dragging and dropping them to different positions in the tree-structure.

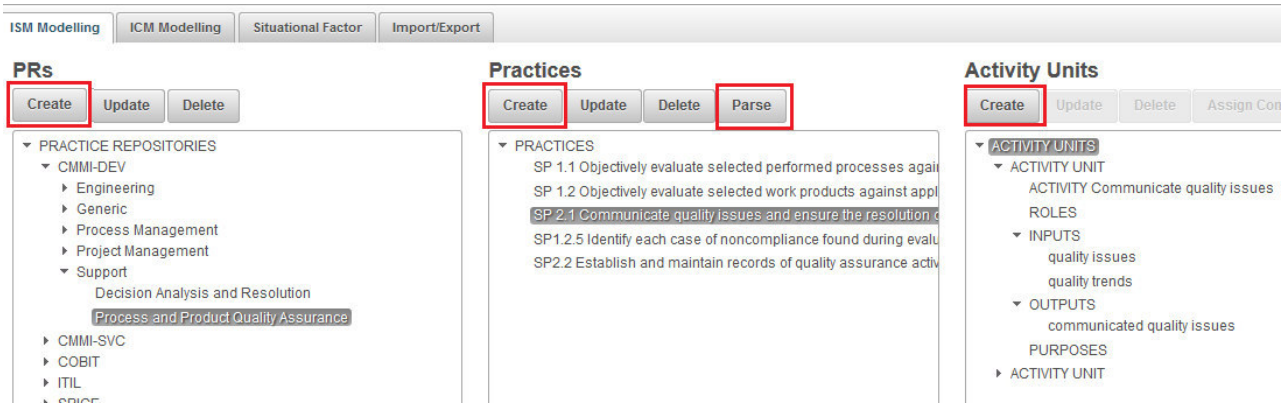


Fig. 59: StepwiseModelerTool – Extract ISM elements – Example

The Modeler extracts the ICM concepts by using the `StepwiseModelerTool` (Fig. 60):

- He searches the ICM concept in ICM – Search by Hierarchy or Search by Name.
- He creates the ICM concept and its ICM abstract concept – Create child and Create abstract respectively. The similarity relation ICM generalizationOf is also created when Create child action is performed.
- He defines the similarity relations ICM composedOf between the created ICM concept and existing ICM concepts by selecting and relating them – Assign composition. The similarity relations ICM generalizationOf and composedOf between the ICM concepts are visualized in the Generalization and in the Composition list respectively.

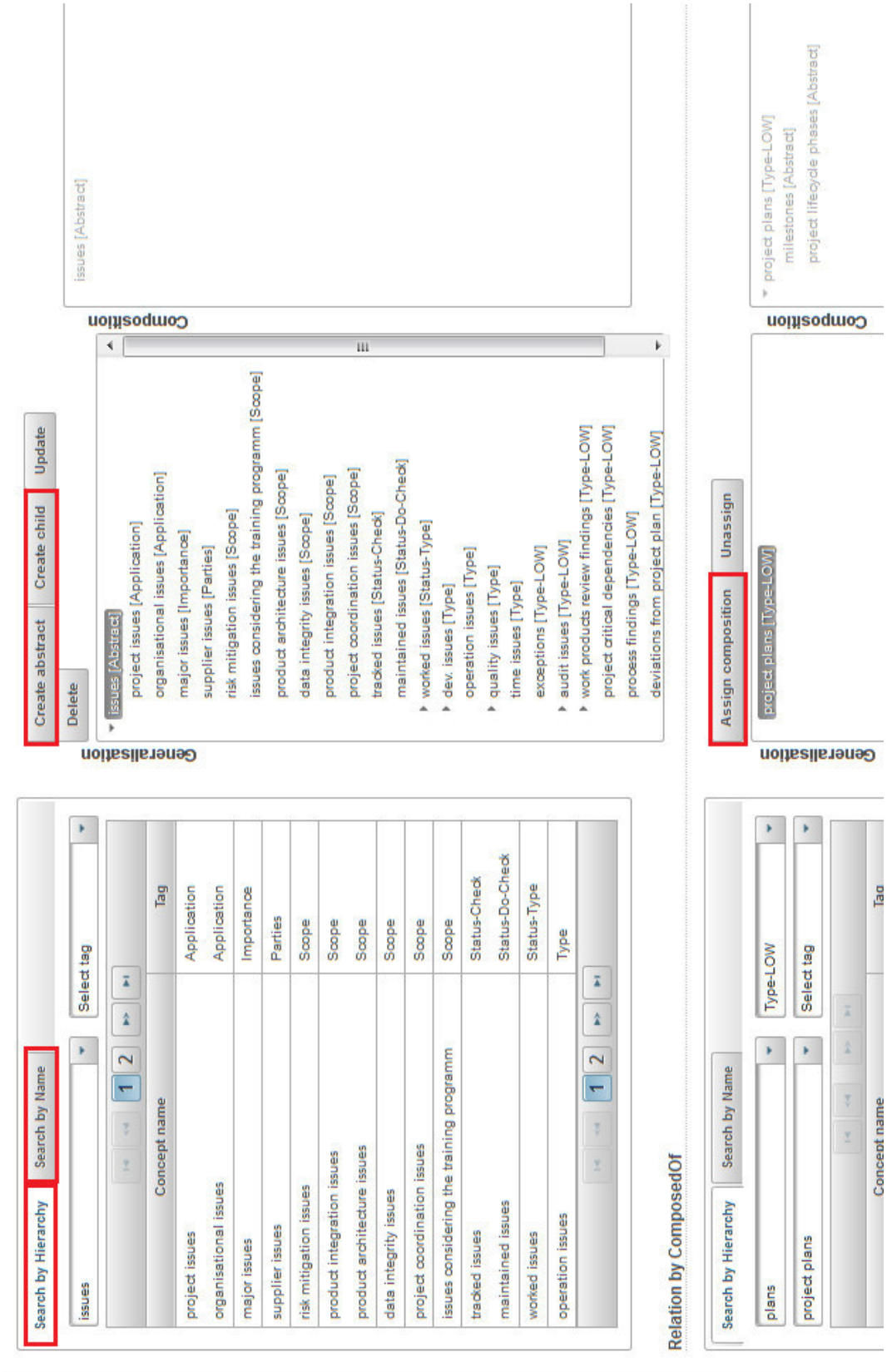


Fig. 60: StepwiseModelerTool – Extract ICM concepts – Example

Once the ICM concepts are extracted, the Modeler can relate them to ISM practiceConcepts – Assign concept (Fig. 61).

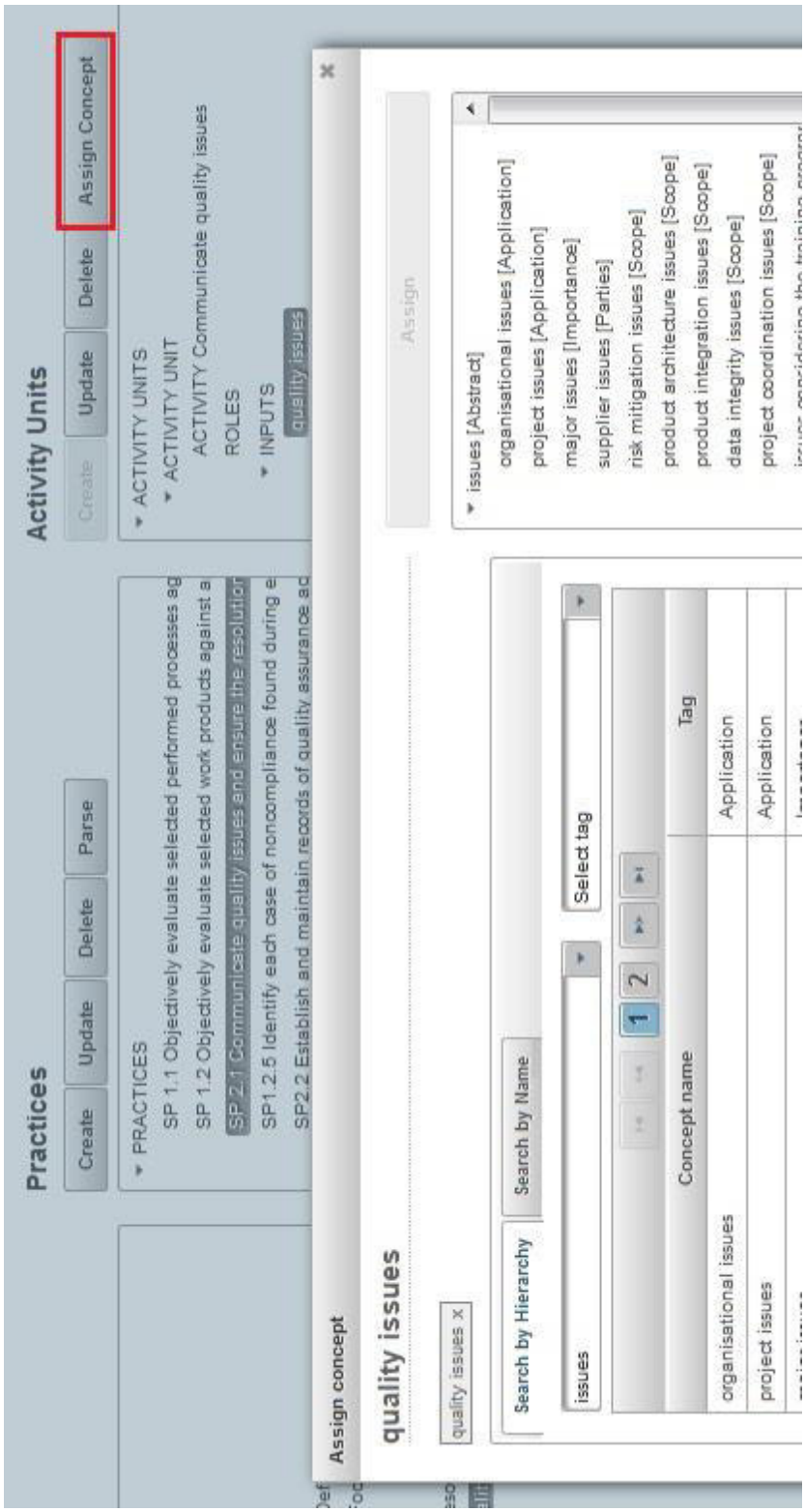


Fig. 61: StepwiseModelerTool – Relate ICM concepts to ISM practiceConcepts – Example

Finally, the Modeler extracts the SFM situationalFactors (Create) and relates them to the ICM concepts (Create relation) by using the `StepwiseModelerTool` (Fig. 62)

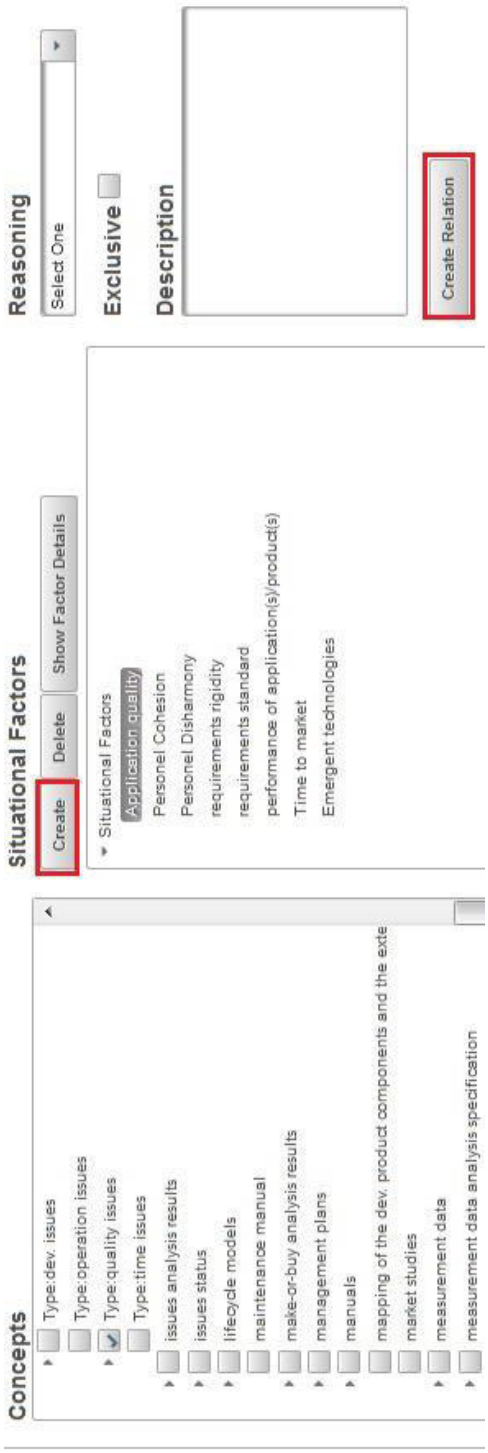


Fig. 62: StepwiseModelerTool – Extract SFM situationalFactors and relate them to ICM concepts – Example

10.2.2.2 XMLModelerTool

The Modeler uses `XMLModelerTool` to model the data in XML format and import it into the database (Fig. 68).

10.2.2.2.1 TextEditorTool

The Modeler can use the `Notepad` or any other text editor to extract ISM elements, relate ISM practiceConcepts to ICM concepts, extract SFM situationalFactors and relate them to ICM concepts. As this tool allows to enter the data in each possible format, we introduce an XML-Schema to define the accepted XML format. We call this schema the TextEditorTool-Schema. There are various tools to represent such XML Schema (oXygen[20], Alt[21], Sty[22]). We use oXygen to represent the TextEditorTool-Schema. Fig. 63 illustrates a legend of the used oXygen XML Schema elements.

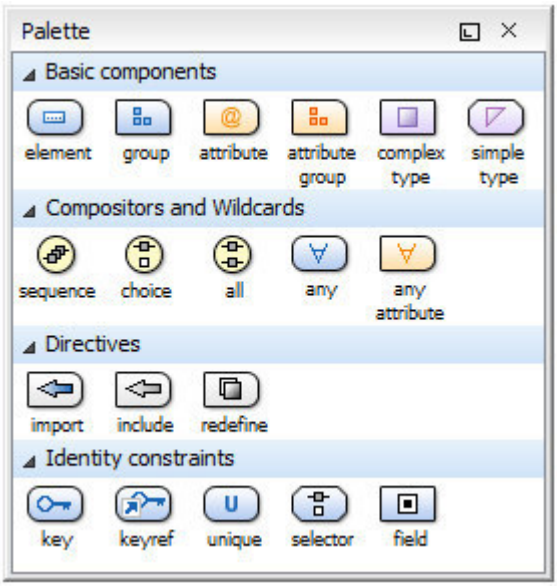


Fig. 63: oXygen XML Schema elements

The TextEditorTool-Schema is based on the MOSAIC meta-models and thus on the elements' and their relations' names (Fig. 64).

[20] OXygen available at http://www.oxygenxml.com/xml_schema_editor.html
[21] Alt available at http://www.eclipse.org/m2m/atl/
[22] Sty available at http://www.stylusstudio.com/xml_schema_editor.html

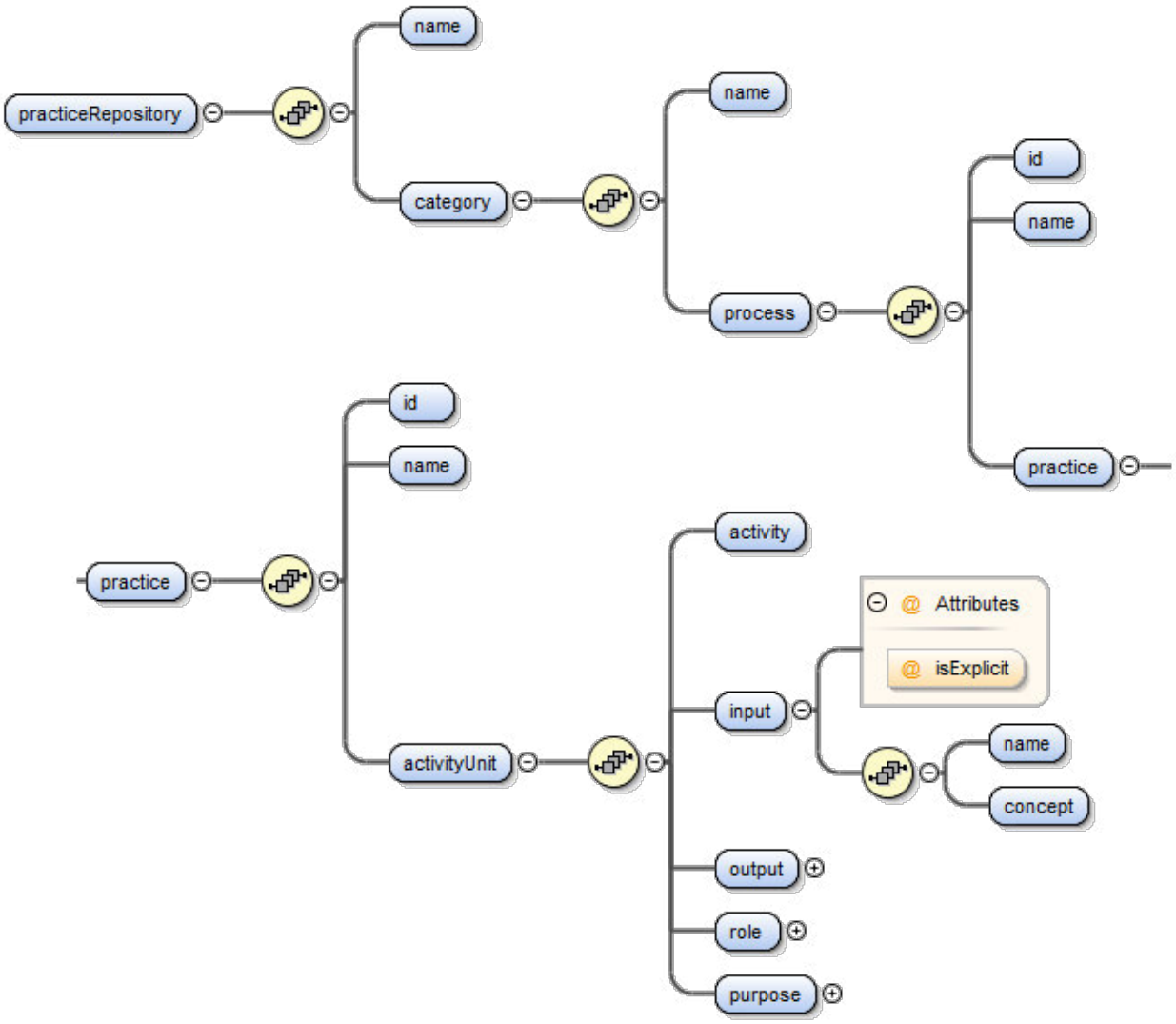


Fig. 64: TextEditorTool-Schema for the extraction of ISM elements

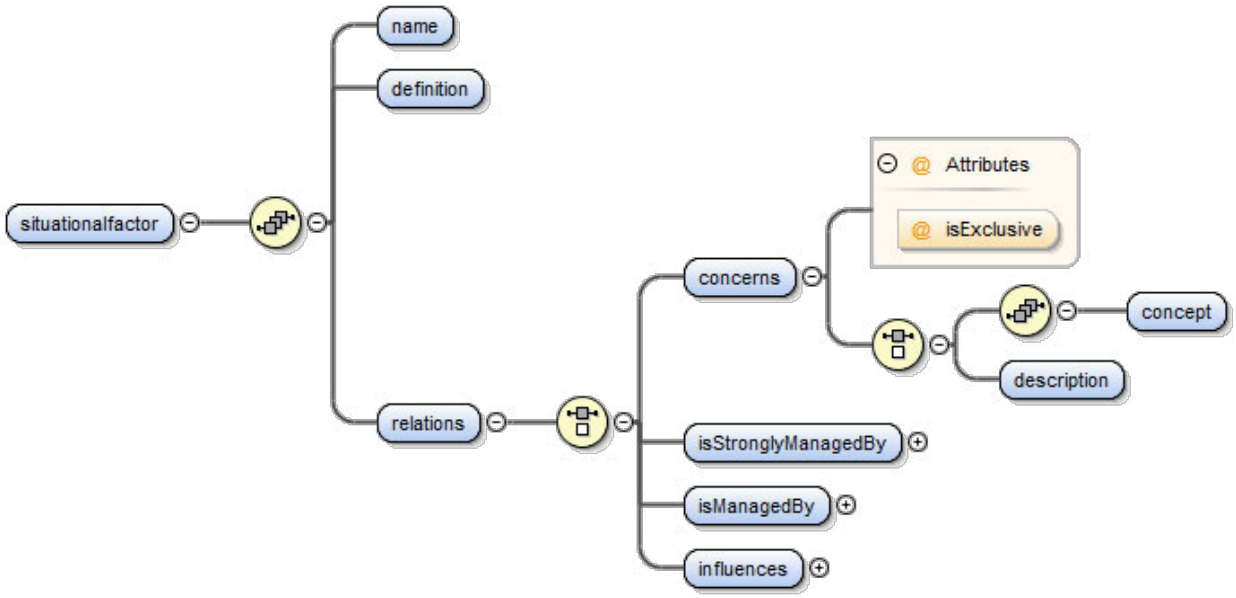


Fig. 65: TextEditorTool-Schema for the extraction of SFM elements

10.2.2.2.2 GraphEditorTool

The Modeler can use the `yEDGraphEditorTool`[23] to extract ICM concepts based on ISM practiceConcepts. As the name says, the `yEDGraphEditorTool` is a graph editor where graph elements and their relations can be edited. There are different ways to represent such elements and relations, for example by using rectangles or arrows. Therefore, we introduce a schema to define which representations are accepted by MOSAIC (Fig. 66).

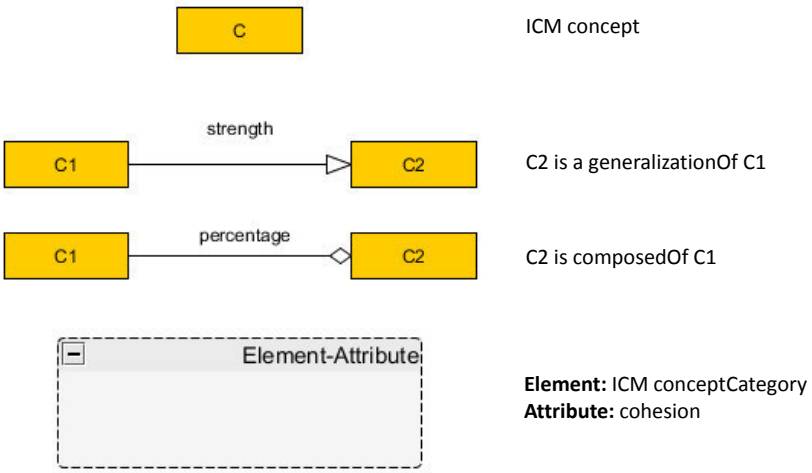


Fig. 66: yEDGraphEditor – Elements and relations accepted by MOSAIC

[23] yEDGraphEditor at http://yworks.com

The output of the `yEDGraphEditorTool` is a graphml XML file that can be imported by the `XMLImporterTool` in the database.

Fig. 67 illustrates an excerpt of the ICM and exemplifies the generalizationOf-mono-hierarchy of the ICM concept “risks”.

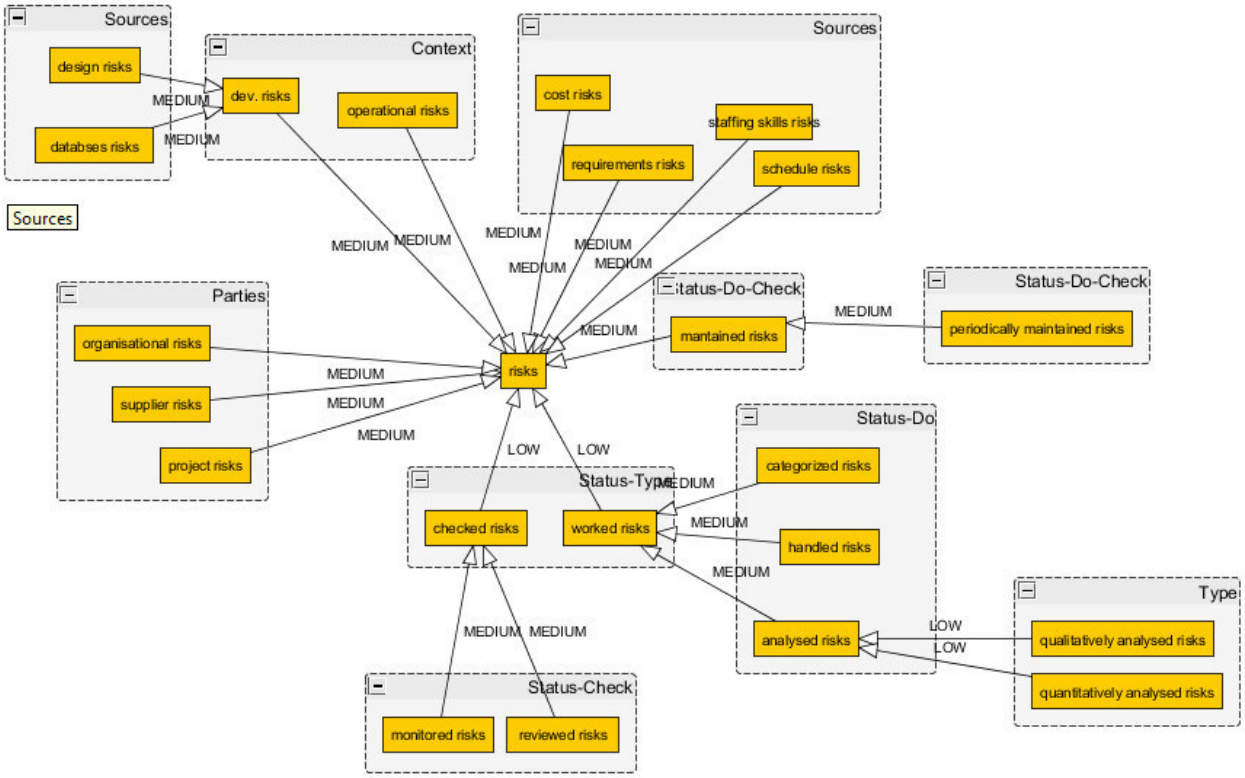


Fig. 67: Excerpt of the ICM

10.2.2.2.3 XMLImporterTool

The Modeler uses the `XMLImporterTool` to import the data in XML format that he models using the `TextEditorTool` and the `GraphEditorTool`. Fig. 68 exemplifies the upload of the ICM, ISM and SFM that are contained in the ICM.graphml, ISM.xml and SFM.xml files.

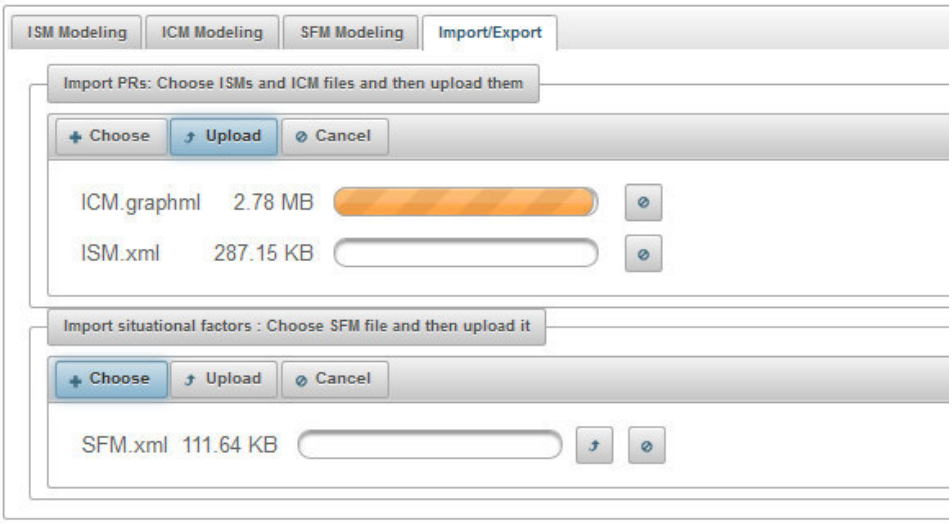


Fig. 68: XMLImporterTool – Import ISMs, ICM and SFM elements – Example

10.2.3 Analyzer Tools

An Analyzer uses the analyzer tools to select ISM practices from multiple PRs based on a SFM situationalFactor, to identify the similar ISM practices as well as to identify the ISM practice dependencies from multiple PRs.

10.2.3.1 SelectionTool

The Analyzer uses the `SelectionTool` to select ISM practices based on SFM situationalFactors. He selects an SFM situationalFactor and the ISM practiceRepositories he is interested in. The `SelectionTool` visualizes the corresponding ISM practices with their support degree. Fig. 69 illustrates the selection of the ISM practices CMMI-DEV PPQA SP2.1 and of SPICE SUP.1 BP9 from the running example.

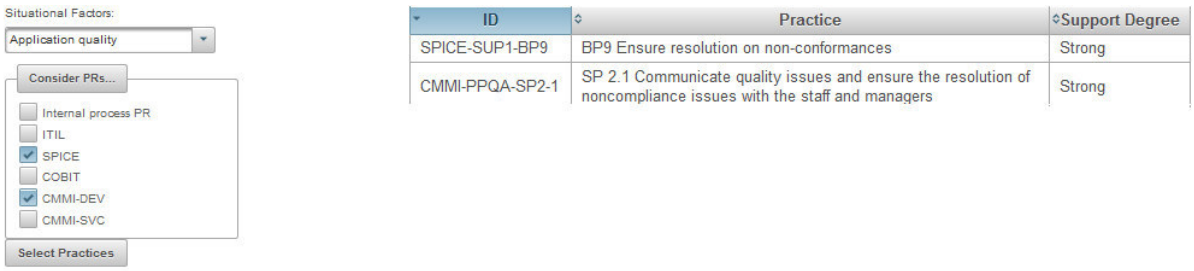


Fig. 69: SelectionTool – ISM Practice Selection – Example

10.2.3.2 SimilarityTool

The Analyzer uses the `SimilarityTool` to identify similar ISM practices. There are three possibilities to identify similar ISM practices.

Firstly, the Analyzer computes the *similarity degree of the ISM practices*. For this purpose, he has to select ISM practices or ISM processes. If ISM processes are selected then all their ISM prac-

tices are considered. The PRs are visualized as a tree, where each element can be expanded to vizualize its sub-elements: an ISM practiceRepository with its ISM categories, ISM categories with its ISM processes, and ISM processes with its ISM practices (Fig. 70).

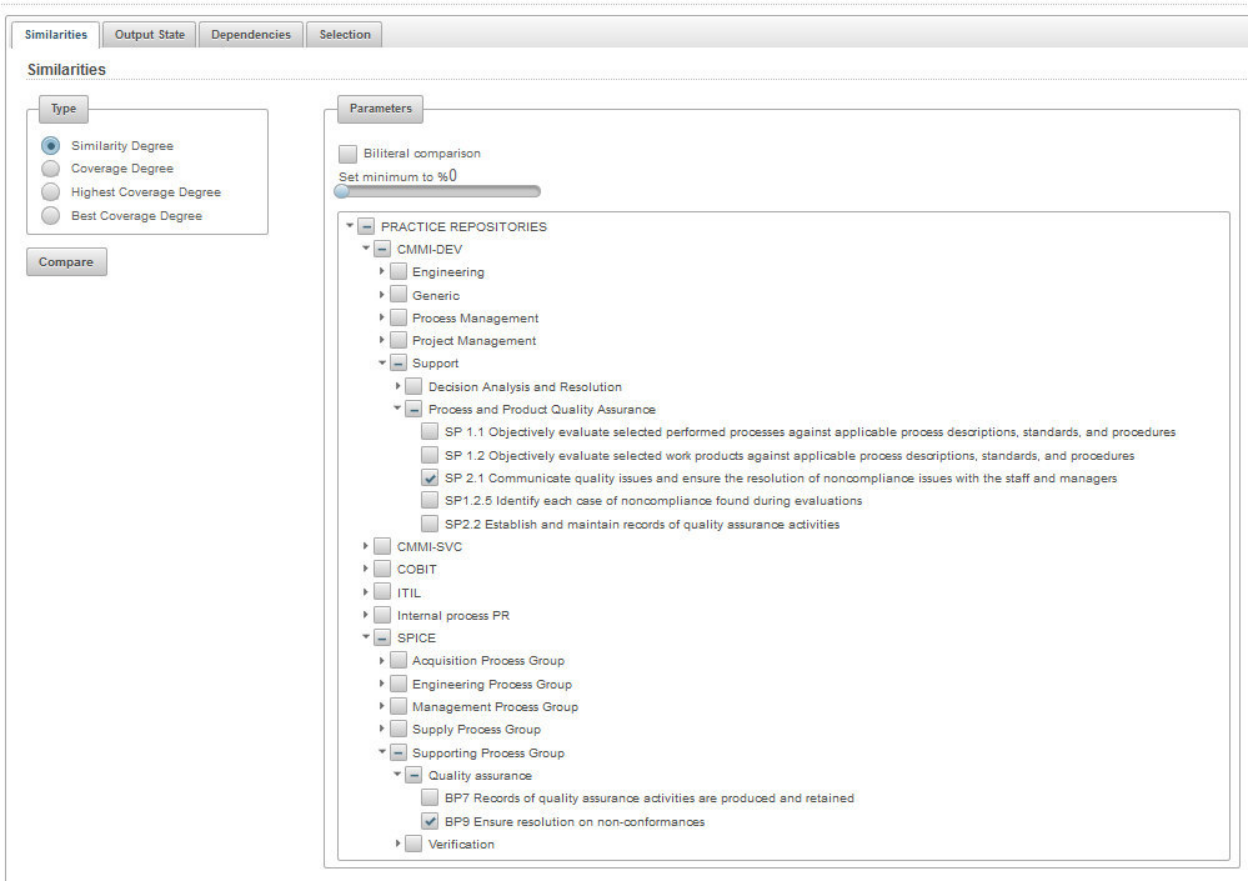


Fig. 70: SimilarityTool – Selection of ISM practices in the PRs tree – Example

The Analyzer can choose between the following two options to compute the similarity degree:

- *Bilateral comparison*: pairs of two ISM practices are generated and compared.
- *Multiple comparisons*: all ISM practices are simultaneously compared.

Furthermore, a minimum threshold for the similarity degree can also be set to filter the results. Finally, the Analyzer initiates the comparison (Compare).

The application displays an overview of similarities between the considered ISM practices (Fig. 71). The maximum similarity degrees of the ISM activityUnits within all possible combinations of ISM activitiyUnits of the considered ISM practices are displayed.

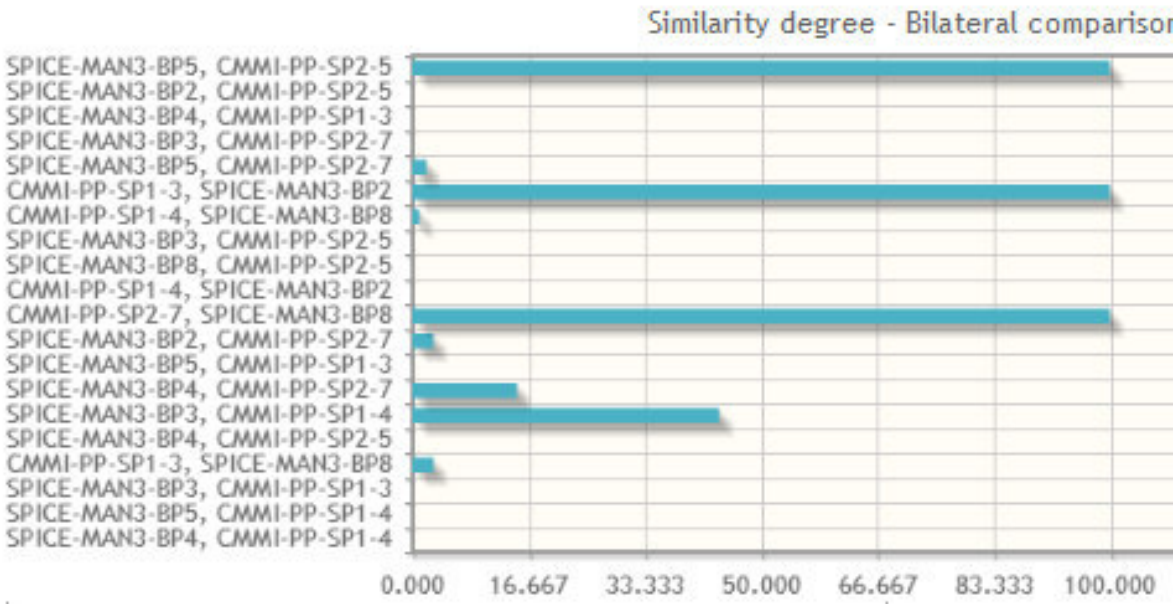


Fig. 71: SimilarityTool – Similarity degree overview – Example

A more detailed view with the similarity degrees between the ISM activityUnits, their ISM outputs, inputs, roles and purposes are also visualized (Fig. 72). For example, the similarity degree of the ISM activityUnits "Ensure resolution of non-conformances" and "Ensure the resolution of non-compliance issues with the staff and managers" of the CMMI-DEV and SPICE practices is 0.83 and thus, is "High". Furthermore, the tool also visualizes the ICM abstract concepts that correspond to the ISM practiceConcepts of the considered ISM practices.

Units	Result	Abstract concept
▾ HIGH		
▾ CMMI-PPQA-SP2-1, SPICE-SUP1-BP9		
▾ | Ensure resolution on non-conformances | Ensure the resolution of noncompliance issues with the staff and managers	0.83	
▾ Role		
managers; staff	0.0	stakeholders
▾ Input		
non-conformances; noncompliance issues	1.0	non-compliance issues
▾ Output		
resolved non-conformances; resolved noncompliance issues	1.0	closed issues; non-compliance issues
Purpose		
▸ | Communicate quality issues | Ensure resolution on non-conformances	0.54	

Fig. 72: SimilarityTool – Similarity degree results – Example

Secondly, the Analyzer can compute the *coverage degree* of ISM practices. Analogously to the similarity degree, he selects ISM practices or ISM processes to perform this activity. For example, the coverage degree between the CMMI-DEV practices and SPICE practices is calculated (Fig. 73). As the coverage degree for the CMMI-DEV practices is 1, the CMMI-DEV practices cover the SPICE practices. The SPICE practices do not entirely cover the CMMI-DEV practices as the coverage degree is 0.75.

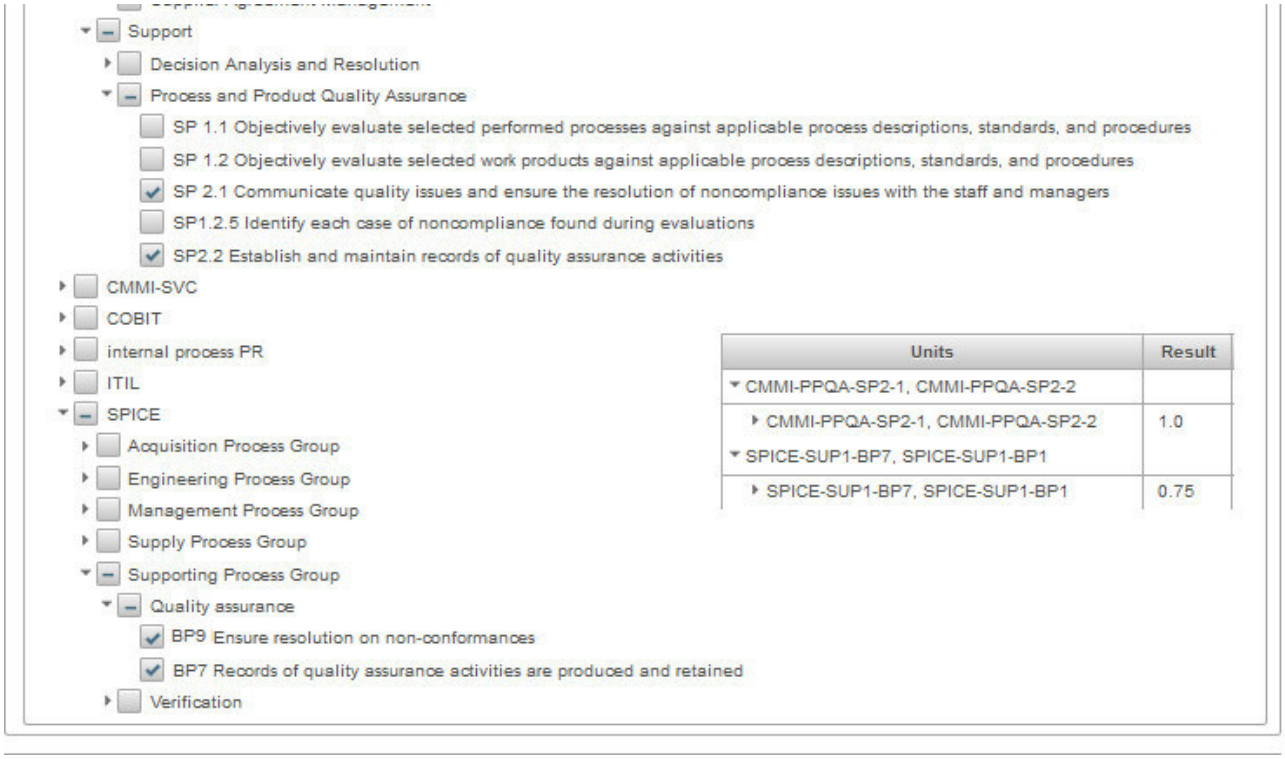


Fig. 73: SimilarityTool – Coverage Degree – Example

He can also calculate the *highest coverage degree or the best coverage for a set of ISM practices*. For example, the highest coverage is given by the CMMI-DEV PPQA SP2.1 practice as this has the maximum coverage degree. The SPICE SUP.1 BP9 practice has a coverage degree of 0.64 in the set of the CMMI-DEV and SPICE practices. Furthermore, additional information is displayed (Fig. 74). The additional information can be:

- *Fraction result*. This indicates how many ISM practiceConcepts from the set of all ISM practiceConcepts are covered by an ISM practice. For example, the 2 ISM outputs of the CMMI-DEV practice entirely covers all 2 ISM outputs of the SPICE and CMMI-DEV practice. The ISM inputs in the SPICE practice only covers 1.56 from the 2 ISM inputs that are contained in the CMMI-DEV and SPICE practice.
- *Different, common and similar Concepts*. This visualizes the ICM concepts for a certain type of ISM practiceConcept. For example, for the type ISM role, the SPICE practice does not contain the ICM concepts “managers” and “staff”.

Units	Result	Fraction Result	Different Concepts	Common Concepts	Similar Concepts
▾ CMMI-PPQA-SP2-1, SPICE-SUP1-BP9					
▾ SP 2.1 Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers	1.0				
Role	1.0	2.0/2		staff; managers;	
Output	1.0	2.0/2		communicated quality issues; resolved noncompliance issues;	
Input	1.0	2.0/2		quality issues; non-conformances;	
▾ BP9 Ensure resolution on non-conformances	0.64				
Input	0.78	1.56/2	;	non-conformances;	; quality issues
Role	0.0	0.0/2	managers; staff;		
Output	0.77	1.53/2	;	resolved noncompliance issues;	; quality issues; communicated issues

Fig. 74: SimilarityTool – Highest Coverage Degree – Example

Finally, the Analyzer can select an ICM concept to identify the output states of the ISM practices that contain ISM practiceConcepts related to this ICM concept or to ICM similar concepts. The similar ISM practices and their output state are visualized. Fig. 39 illustrates the output states of the CMMI-DEV practices from our running example.

issues | Type
quality issues | Select tag

Practices	State
CMMI-PPQA-SP2-1 SP 2.1 Communicate quality issues and ensure the resolution of noncompliance issues with the staff and managers	Do
CMMI-PPQA-SP1-2-5 SP1.2.5 Identify each case of noncompliance found during evaluations	Plan

Fig. 75: SimilarityTool – Output States – Example

10.2.3.3 DependenciesTool

The Analyzer uses the `DependenciesTool` to identify the dependencies between ISM practices and their dependency degrees. First, he can select the ISM practiceRepositories to specify which PRs have to be considered. Then, he can select ISM processes. The dependencies between all ISM practices of these ISM processes and all ISM practices contained in the selected PRs are computed (Fig. 76). Furthermore, he can check “Equal dependencies”, if he is interested in the dependencies with a “Strong” dependency degree. Otherwise the dependencies with a “Medium” dependency degree are considered

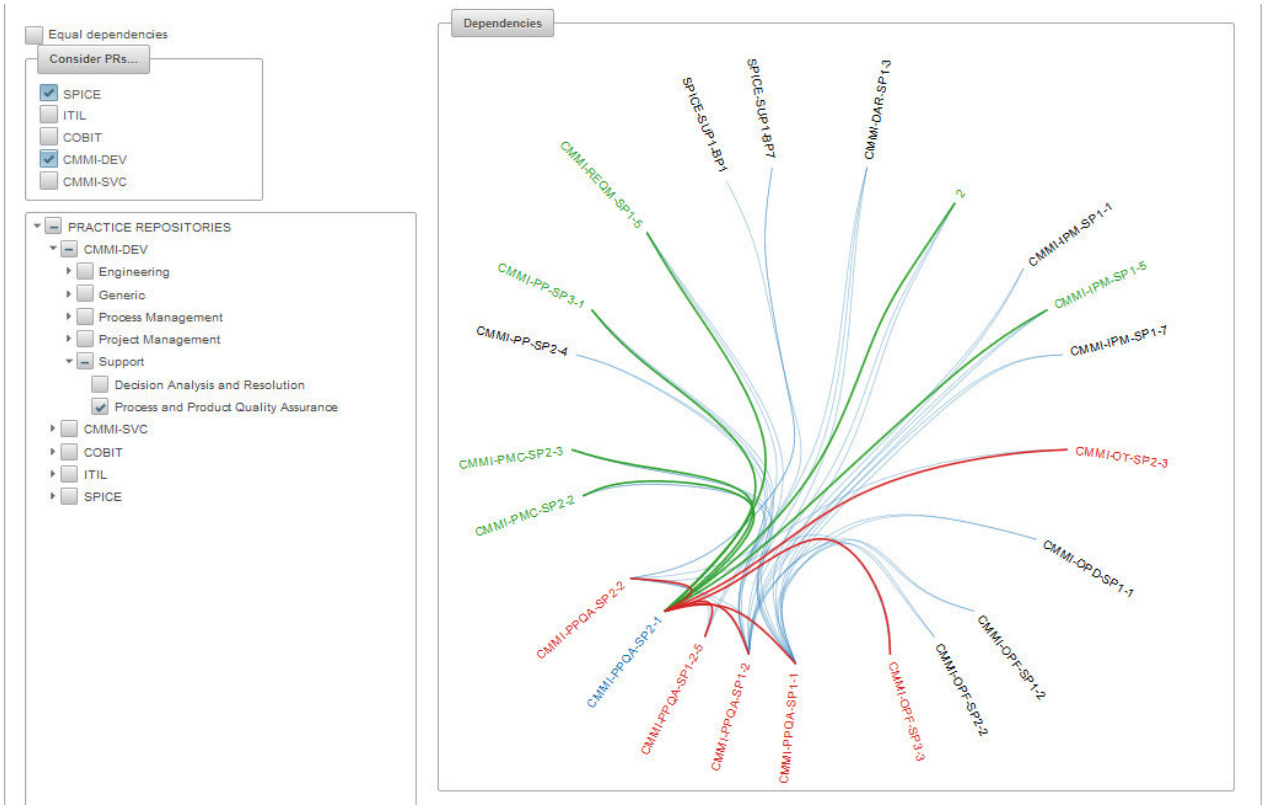


Fig. 76: DependencyTool – Dependencies – Example

11 Acronyms

Acronym	Description
ARAMIS	ARchitecture Analysis and Monitoring InfraStructure
ARAMIS CIC	ARAMIS Communication Integrity Checker
CMMI	Capability Maturity Model Integration
CMMI-DEV	Capability Maturity Model Integration for Development
CMMI-DEV CM	CMMI-DEV Configuration management
CMMI-DEV IPM	CMMI-DEV Integrated Project Management
CMMI-DEV PP	CMMI-DEV Project Planning
CMMI-DEV PPQA	CMMI-DEV Product and Process Quality Assurance
CMMI-DEV REQM	CMMI-DEV Requirements Management
CMMI-DEV VER	CMMI-DEV Verification
CMMI-SVC	Capability Maturity Model Integration for Services
CMMI-SVC WP	CMMI-DEV Work Planning
COBIT	Control Objectives for Information and Related Technology
ICM	Integrated Concept Model
IC Meta-Model	Integrated Concept Meta-Model
ISM	Integrated Structure Model
IS Meta-Model	Integrated Structure Meta-Model
ITIL	Information Technology Infrastructure Library
PRs	Practice Repositories
SFM	Situational Factors Model
SF Meta-Model	Situational Factors Meta Model
SPICE	Software Process Improvement and Capability Determination
SPICE BP	SPICE Best Practice
SPICE ENG	SPICE Engineering
SPICE ENG.6	SPICE ENG Software Construction
SPICE SUP	SPICE Support
SPICE SUP.1	SPICE SUP Quality assurance
SPICE SUP.10	SPICE SUP Change request management
SPICE SUP.2	SPICE SUP Verification
COBIT PO9	COBIT PO – Assess and Manage IT Risks
COBIT PO	COBIT Plan and Organize

12 Dictionary

Term	Definition
Adoption of a PR	An adoption of a PR is the process of following this PR by an organization in order to improve its software processes. The adoption does not only mean the organization-wide process improvement to define an internal process PR, but also the adoption of this PR in software projects.
Assessment based on a PR	An assessment based on a PR is the evaluation process of software processes of an organization or part of an organization according to a PR to identify deviations from this PR and thus, to identify improvement potential for the software processes. If such an assessment is successful, we say that the organization is PR compliant.
Best practice	A best practice (in short practice) describes an activity which has proven itself as a guideline for the improvement of software processes in an organization.
ComposedOf-poly-hierarchy	In a composedOf-poly-hierarchy, an ICM part concept can have more ICM whole concepts.
Domain	A domain refers to a sector within the economic system that provides goods and services for the end-customer, e.g. health care or the automotive industry.
Extraction of ISM and ICM elements	The extraction of ISM or ICM elements consists of the identification of these elements in the PRs as well as their modeling according to the guidelines defined by the MOSAIC meta-models.
Extraction of SFM elements	The extraction of SFM elements consists of the identification of these elements in the situational factors framework [Clarke and O'Connor 2012], as well as their modeling according to the guidelines defined by the MOSAIC meta-models.
GeneralizationOf-mono-hierarchy	In a generalizationOf-mono-hierarchy, each ICM special concepts can have only one ICM general concept.
ICM abstract concept	An ICM abstract concept is a ICM general concept and is the root of the generalizationOf-mono-hierarchy.
ICM ancestor concept	An ICM ancestor concept for an ICM concept conc is a ICM general concept on the single path between the ICM concept conc and the root of the generalizationOf-mono-hierarchy.
ICM child concept	An ICM child concept for a ICM general concept is its ICM special concept for an ICM generalizationOf.
ICM composedOf	ICM composedOf is a similarity relation between an ICM whole concept and an ICM part concept.
ICM concept	An ICM concept is an ICM element, a word or the smallest combination of words contained in a practice that has a unique meaning in the context of PRs (e.g. "project plan", "work breakdown structure", "key stakeholder" or "software stakeholder"). One or more ICM concepts semantically define one ISM practiceConcept contained in different ISMs (e.g. concepts "key stakeholder" and "software stakeholder" define the role "software key stakeholder").

ICM conceptCategory	ICM conceptCategory categorizes different ICM similar concepts in the generalizationOf-hierachy in ICM.
ICM descendant concept	An ICM descendant concept for an ICM concept conc is a ICM special concept on the many paths between the ICM concept conc and ICM concepts on lower levels in the generalizationOf-mono-hierarchy.
ICM different concept	An ICM different concept for other ICM concept is an ICM concept that is not connected with this other ICM concept by any similarity relations.
ICM generalizationOf	ICM generalizationOf is a similarity relation between a ICM general concept and a ICM special concept.
ICM parent concept	An ICM parent concept for a ICM special concept is its ICM general concept for an ICM generalizationOf.
ICM sibling concepts	An ICM sibling concepts are ICM special concepts that share the same ICM general concept for an ICM generalizationOf.
ICM similar concepts	ICM similar concepts are related by ComposedOf or GeneralizationOf.
ICM synonym concepts	ICM synonym concepts have the same sematic meaning and thus, they are semantically equal and refer to the same ICM concept.
Internal process PR	An internal process PR is a special process PR and refers to the internal software processes of an organization. This PR defines activities to be used as guidelines for the improvement of software processes in an organization, e.g. in software projects. Hence, it defines the practices of this organization and thus, it is a PR.
ISM activity	An ISM activity describes an action that is defined by an ISM practice.
ISM activityUnit	An ISM activitiyUnit refers to a group an ISM activity with its related ISM outputs, inputs, roles and purposes.
ISM artifact	An ISM artifact refers to work products that are needed or produced by an ISM activity.
ISM category	An ISM category defines a certain topic that is addressed in one or more ISM processes.
ISM input	An ISM inputs is an ISM artifacts that is needed by an ISM activity.
ISM output	An ISM output is an ISM artifacts that is produced by an ISM activity.
ISM practice element	An ISM practice element refers to the elements contained in an ISM practice.
ISM practice language elements	ISM practice language elements defines syntactical elements that are used to textually describe the ISM practice elements.
ISM practice repository elements	ISM practice repository elements are the elements that group or refer to the ISM practices of the multiple PRs. Such elements are ISM practiceRepositories, categories, processes and practices.
ISM practiceConcept	An ISM practiceConcept is an ISM element, a word or a combination of words which semantically represent the following practice elements: • role, which performs the activity described by the practice. • output, which is produced by this activity. • input, which is needed by this activity to produce the output. • purpose, which motivates the activity operation.
ISM practiceRepository	ISM practiceRepository represents a certain PR.

ISM process	An ISM process addresses a topic to be improved by defining ISM practices.
ISM purpose	An ISM purpose refers to the goal that need to be achieved when performing an ISM activity.
ISM role	An ISM role refers to the role involved in an ISM activity.
Ontology	An ontology is an explicit specification of a conceptualization, an abstract view of the world that we wish to represent for a purpose. Therefore, it is an linguistic, formal representation of a set of concepts and their relations within a particular domain.
Practice repository	A practice repository (in short PR) is a collection of practices defined as being based on the experience and knowledge of many practitioners over the years in order to be used for the improvement of software processes in an organization. A practice repository does not only contain practices, but also other elements that are related to these practices (e.g. processes or activities are such elements).
Process PR	A process PR refers to a process model and is more concrete than a reference PR be-cause it defines not only practices, but also gives additional information about how to adopt these practices. It can be used as a guideline, but it can also be directly applied to describe the software processes of an organization. For example, the V-Model XT is such a process PR.
Reference PR	A reference PR is used as a guideline for the software process improvement of organizations. For example, CMMI-DEV, COBIT or ISO/IEC 12207 are reference PRs.
Schema-based matching	Schema-based matching is the process of identification of elements that are semantically related.
SFM situationalFactor	A SFM situationalFactor is a situational factor defined as “a characteristic of a software development setting that is known to affect the software development"[Clarke and O’Connor 2012].
Software area	A software area is a sphere of activities in an organization related to a software product.

13 Bibliography

ABRAHAMSSON, P., CONBOY, K., AND WANG, X. 2009. “Lots done, more to do”: the current state of agile systems development research. 18, 4, 281–284. http://goo.gl/5xSnBf.

ABRAN, A. AND BUGLIONE, L. 2008. Assessment of Measurement Indicators in Software Process Improvement Frameworks. http://goo.gl/rBZnJO.

ALBRECHT, A.J. 1979. Measuring application development productivity. Proceedings of the IBM Applications Development Symposium, 83–92. http://goo.gl/wFL2k4.

ALISCH, K. 2005. Gabler Wirtschaftslexikon. Gabler. http://goo.gl/t2e81U.

ANDELFINGER, U., HEIJSTEK, A., AND KIRWAN, P. 2006. A Unified Process Improvement Approach for Multi-Model Improvement Environments. Software Engineering Institute, Carnegie Mellon. http://goo.gl/yzPuZj.

AXELOS. 2011. Service Management - ITIL® 2011 Edition Publications. http://goo.gl/f2tnRZ.

BAADER, F., BERNARDI, R., CALVANESE, D., ET AL. 2007. Techniques for Ontology Design and Maintenance. http://goo.gl/GHU2ZE.

BALDASSARRE, M.T., PIATTINI, M., PINO, F.J., AND VISAGGIO, G. 2009. Comparing ISO/IEC 12207 and CMMI-DEV: Towards a Mapping of ISO/IEC 15504-7. Proceedings of the Seventh ICSE Conference on Software Quality, IEEE Computer Society, 59–64. http://goo.gl/s07dG1.

BASILI, V.R. 1992. Software Modeling and Measurement: The Goal/Question/Metric Paradigm. University of Maryland at College Park, College Park, MD, USA. http://goo.gl/fZEJLT.

BASILI, V.R., LINDVALL, M., REGARDIE, M., ET AL. 2010. Linking Software Development and Business Strategy Through Measurement. Computer 43, 4, 57–65. http://goo.gl/Sx4iOw.

BASILI, V.R. AND ROMBACH, H.D. 1988. The TAME project: towards improvement-oriented software environments. IEEE Transactions on Software Engineering 14, 6, 758–773. http://goo.gl/t9rM6X.

BASILI, V.R. AND WEISS, D.M. 1984. A Methodology for Collecting Valid Software Engineering Data. IEEE Trans. Software Eng. 10, 6, 728–738. http://goo.gl/YYYWLm.

BECK, K. 2000. Extreme Programming Explained: Embrace Change. Addison-Wesley, Reading, MA.

BENEDIKTSSON, O., DALCHER, D., AND THORBERGSSON, H. 2006. Comparison of software development life cycles: a multiproject experiment. IEE Proceedings - Software 153, 3, 87. http://goo.gl/hjy0RJ.

BHUTA, J., BOEHM, B., AND MEYERS, S. 2005. Process Elements: Components of Software Process Architectures. Proceedings of the 2005 International Conference on Unifying the Software Process Spectrum, Springer-Verlag, 332–346. http://goo.gl/RY4JfT.

BIRD, S. 2009. Natural Language Processing with Python. O'Reilly.

BOEHM, B.W. 1981. Software Engineering Economics. Prentice-Hall. http://goo.gl/B5NCId.

BOEHM, B.W. 1991. Software risk management: principles and practices. 8, 1, 32–41. http://goo.gl/6kHHCy.

BOEHM, B.W., ABTS, C., WINSOR BROWN, A., ET AL. 2000. Software cost estimation with Cocomo II. Prentice Hall, Upper Saddle River, NJ. http://goo.gl/gdARUF.

BOEHM, B.W. AND TURNER, R. 2004. Balancing agility and discipline: a guide for the perplexed. Addison-Wesley, Boston. http://goo.gl/m8y5sr.

BOWERS, J., MAY, J., MELANDER, E., BAARMAN, M., AND AYOOB, A. 2002. Tailoring XP for Large System Mission Critical Software Development. In: D. Wells and L. Williams, eds., Extreme Programming and Agile Methods — XP/Agile Universe 2002. Springer, 100–111. http://goo.gl/pzofOK.

BRESCIANI, P., PERINI, A., GIORGINI, P., GIUNCHIGLIA, F., AND MYLOPOULOS, J. 2004. Tropos: An Agent-Oriented Software Development Methodology. Autonomous Agents and Multi-Agent Systems 8, 3, 203–236. http://goo.gl/K1vyW0.

BRIAND, L.C., DIFFERDING, C.M., AND ROMBACH, H.D. 1996. Practical Guidelines for Measurement-Based Process Improvement. 100–113. http://goo.gl/bpZ2k0.

BUCH, C. 2010. ISO 56K – Einführung einer dreifachen ISO – Zertifizierung in nur zwei Jahren. 8. meetIT!L Service Management Kongress 2010. http://goo.gl/GKBy88.

BUGLIONE, L. 2008. Strengthening CMMI Maturity Levels with a Quantitative Approach to Root-Cause Analysis. Proceedings of the 5th Software Measurement European Forum (SMEF 2008), Milan, Italy, May, 28–30. http://goo.gl/FqfQwi.

BULLEN, C.V. AND ROCKART, J.F. 1981. A primer on critical success factors. Massachusetts Institute of Technology (MIT), Sloan School of Management Series Working papers, 1220. http://goo.gl/Vey2FP.

CAO, L., MOHAN, K., XU, P., AND RAMESH, B. 2004. How extreme does extreme programming have to be? Adapting XP practices to large-scale projects. Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04), 30083.3. http://goo.gl/9AHIiN.

CATER-STEEL, A. AND ROUT, T. 2008. SPI Long-Term Benefits: Case Studies of Five Small Firms. In: Software Process Improvement for Small and Medium Enterprises - Techniques and Case Studies. Hanna Oktaba and Mario Pattini, 223–241. http://goo.gl/jHm4D6.

CEPEDA, S., GARCIA, S., AND LANGHOUT, S. 2008. Is CMMI Useful and Usable in Small Settings? One Example. CrossTalk, The Journal of Defense Software Engineering 21, 2, 14–18. http://goo.gl/ibmNr7.

CHEN, X. AND STAPLES, M. 2007. Using Practice Outcome Areas to Understand Perceived Value of CMMI Specific Practices for SMEs. EuroSPI'07 Proceedings of the 14th European conference on Software Process Improvement, Potsdam, Germany, Springer, 59–70. http://goo.gl/bMLljG.

CHEN, X., STAPLES, M., AND BANNERMAN, P. 2008. Analysis of Dependencies between Specific Practices in CMMI Maturity Level 2. EuroSPI'08 Proceedings of the 15th European conference on Software Process Improvement, Dublin, Ireland, Springer Berlin Heidelberg, 94–105. http://goo.gl/9yhDKs.

CLARKE, P. AND O'CONNOR, R.V. 2012. The situational factors that affect the software development process: Towards a comprehensive reference framework. 54, 5, 433–447. http://goo.gl/xu2Ns9.

CMMI PRODUCT TEAM. 2010a. CMMI for Development, Version 1.3 (CMU/SEI-2010-TR-033). Software Engineering Institute, Carnegie Mellon University. http://goo.gl/qlz7XU.

CMMI PRODUCT TEAM. 2010b. CMMI for Services, Version 1.3 (CMU/SEI-2010-TR-034). Software Engineering Institute, Carnegie Mellon University. http://goo.gl/IauZPL.

COCKBURN, A. 2000. Selecting a project's methodology. 64–71. http://goo.gl/YNOpV6.

COLEMAN, G. AND O'CONNOR, R. 2008. Investigating software process in practice: A grounded theory perspective. Journal of Systems and Software 81, 5, 772–784. http://goo.gl/SdSXkX.

CONBOY, K. 2009. Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development. Information Systems Research 20, 3, 329–354. http://goo.gl/b1yRe4.

CONRADI, H. AND FUGGETTA, A. 2002. Improving software process improvement. Software, IEEE 19, 4, 92–99. http://goo.gl/Ii80OF.

CUNNINGHAM, H., MAYNARD, D., BONTCHEVA, K., ET AL. 2012. Developing language processing components with gate (a user guide). Version 7. http://goo.gl/9F9iyL.

CURTIS, B., KELLNER, M.I., AND OVER, J. 1992. Process Modeling. Communications of the ACM 35, 9, 75–90. http://goo.gl/DmXw5g.

DEDE, B. AND LIOUFKO, I. 2010. Situational Factors Affecting Software Development Process Selection. Master of Science, University of Gothenburg. http://goo.gl/n22yEh.

DEKKERS, C. 2013. Scope Management - Bridging the Gap. http://goo.gl/aaof5o http://goo.gl/aaof5o.

DKE GREMIUM 914. 1999. IEC 61508. ISO.

DRAGOMIR, A. AND LICHTER, H. 2013. Run-time Monitoring and Real-time Visualization of Software Architectures. Proceedings of the 20th Asia-Pacific Software Engineering Conference, Bangkok, Thailand, 396–403. http://goo.gl/K4tJJb.

DRAGOMIR, A.-M.-C. 2011. A Tool Based Approach for the Extraction of Reference Model Concepts. RWTH Aachen. http://goo.gl/BfhdMO.

DYBA, T. 2005. An empirical investigation of the key factors for success in software process improvement. IEEE Transactions on Software Engineering 31, 5, 410–424. http://goo.gl/8LVYEd.

DYBÅ, T. AND DINGSØYR, T. 2008. Empirical studies of agile software development: A systematic review. Information and Software Technology 50, 9-10, 833–859. http://goo.gl/BFhh7B.

EKERT, D., HAGENMEYER, P., AND MESSNARZ, R. 2009. Development of a concept for integrating various Quality Standards. EuroSPI'09 Proceedings of the 16th European conference on Software Process Improvement, Alcala, Spain, Springer.

FAIRLEY, R.E. AND WILLSHIRE, M.J. 2003. Why the vasa sank: 10 problems and some antidotes for software projects. 18–25. http://goo.gl/7EL1ol.

FAVRE, J.-M. 2005. Megamodelling and Etymology. Transformation Techniques in Software Engineering, 22. http://goo.gl/TU9VnK.

FEILER, P.H. AND HUMPHREY, W.S. 1992. Software process development and enactment: concepts and definitions. IEEE Computer Society Press, 28–40. http://goo.gl/dhrAFk.

FENTON, N.E. AND PFLEEGER, S.L. 1997. Software metrics: a rigorous and practical approach. PWS Pub., Boston.

FERCHICHI, A. AND BIGAND, M. 2008. An Ontology for Quality Standards Integration in Software Collaborative Projects. Proceedings of MDISIS'08 (Model Driven Interoperability for Sustainable Information Systems), Montpellier, France. http://goo.gl/YMYkSm.

FERREIRA, A.L., MACHADO, R.J., AND PAULK, M.C. 2010. Size and Complexity Attributes for Multimodel Improvement Framework Taxonomy. 36th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), Lille, France, 306–309. http://goo.gl/oqyTuQ.

FERREIRA, A.L., MACHADO, R.J., AND PAULK, M.C. 2011. Supporting Audits and Assessments in Multi-model Environments. Proceedings of the 12th International Conference on Product-focused Software Process Improvement, Springer-Verlag, 73–87. http://goo.gl/KhJhJo.

FERRUCCI, D., LALLY, A., GRUHL, D., ET AL. 2006. Towards an Interoperability Standard for Text and Multi-Modal Analytics. RC24122 (W0611-188), Computer Science. http://goo.gl/9ZPSDk.

FOWLER, M. AND HIGHSMITH, J. 2001. The Agile Manifesto. Agile Alliance. http://goo.gl/J0lIY5.

FRANCIS, W.N. AND KUCERA, H. 1979. MANUAL OF INFORMATION to accompany A Standard Corpus of Present-Day Edited American English, for use with Digital Computers. University of Maryland, Department of Computer Science. http://goo.gl/H1CV9w.

FRICKER, S.A., PERSSON, M., AND LARSSON, M. 2013. Tailoring the Software Product Management Framework for Use in a Healthcare Organization: Case Study. EuroSPI '13 Proceedings of the 20th European System, Software & Service Process Improvement & Innovation, Dundalk, Ireland, Springer, 155–166. http://goo.gl/dpNWr4.

GANESAN, P., GARCIA-MOLINA, H., AND WIDOM, J. 2003. Exploiting hierarchical domain structure to compute similarity. Journal ACM Transactions on Information Systems (TOIS) 21, 1, 64–93. http://goo.gl/ZnhON0.

GARCIA, S. 2007. Process Improvement "At the Edges."Webinar, Software Engineering Institute, Carnegie Mellon. http://goo.gl/EkuNJ0.

GELTINGER, G. 2010. Das SOPHIST-REgelwerk Anwendung und Bewertung einer Methode des Requirements Engineerings. GRIN Verlag GmbH, Ravensburg. http://goo.gl/yNV7nZ.

GIBSON, D., GOLDENSON, D.R., AND KOST, K. 2006. Performance Results of CMMI®-Based Process Improvement. Software Engineering Institute, Carnegie Mellon University. http://goo.gl/1yBvRL.

GOLDSTONE, R.L. 1994. Similarity, Interactive Activation, and Mapping. Journal of Experimental Psychology: Learning, Memory, and Cognition 20, 3–28. http://goo.gl/438c9c.

GOLDSTONE, R.L. AND SON, J.Y. 2005. Similarity. In: Cambridge University Press, Cambridge, 13–36. http://goo.gl/KPpsaq.

GÓMEZ ROSENKRANZ, C.A. 2010. Developing a model based approach for integrating improvement instruments. RWTH Aachen. http://goo.gl/ttfUYT.

GOODMAN, N. 1972. Problems and Projects. Hackett Pub. Co., U.S. http://goo.gl/HwhRYK.

GRINDER, J. AND BANDLER, R. 1976. The structure of magic II. Science and Behavior Books, Palo Alto, CA, USA. http://goo.gl/XGizSg.

GROENEN, P.J. AND BORG, I. 2013. Past, Present, and Future of Multidimensional Scaling. Erasmus University Rotterdam, Erasmus School of Economics (ESE), Econometric Institute in its series Econometric Institute Research Papers. http://goo.gl/8Kv343.

GRUBER, T.R. 1993. A Translation Approach to Portable Ontology Specifications. Knowl. Acquis. 5, 2, 199–220. http://goo.gl/fD180B.

HABRA, N., ALEXANDRE, S., DESHARNAIS, J.-M., LAPORTE, C.Y., AND RENAULT, A. 2008. Initiating software process improvement in very small enterprises. Information and Software Technology 50, 7-8, 763–771. http://goo.gl/ARhUfl.

HALVORSEN, C.P. AND CONRADI, R. 2001. A Taxonomy to Compare SPI Frameworks. In: V. Ambriola, ed., EWSPT. Springer, 217–235. http://goo.gl/M0pebr.

HERBSLEB, J.D. AND GOLDENSON, D.R. 1996. A systematic survey of CMM experience and results. IEEE Computer Society Press, 323–330. http://goo.gl/gnrgGg.

HESTON, K.M. AND PHIFER, W. 2011. The multiple quality models paradox: how much 'best practice' is just enough? Journal of Software Maintenance and Evolution: Research and Practice 23, 8, 517–531. http://goo.gl/ifSPTM.

HOFFMANN, V. 2013. Rapid Prototyping in der Use-Case-zentrierten Anforderungsanalyse. Shaker, Herzogenrath.

HÖHN, R. 2008. Das V-Modell XT Anwendungen, Werkzeuge, Standards. Springer. http://goo.gl/6R0KlW.

HOSSAIN, E., BABAR, M.A., AND PAIK, H. 2009. Using Scrum in Global Software Development: A Systematic Literature Review. Proceedings of the Global Software Engineering, ICGSE '09, 175–184. http://goo.gl/Eqzxm5.

HUMPHREY, W.S. 1988. Characterizing the software process: a maturity framework. IEEE Software 5, 2, 73–79. http://goo.gl/WUtAEF.

ISACA. 2008. VAL IT Framework. http://goo.gl/nWIFQD.

ISACA. 2011a. COBIT®5: Process Reference Guide. http://goo.gl/aPYq0o.

ISACA. 2011b. COBIT Mapping: Mapping of CMMI for Development V1.2 With COBIT 4.1. http://goo.gl/16bmrT.

ISACA. 2011c. Global Status Report on the Governance of Enterprise IT (GEIT). http://goo.gl/5dSaJp.

ISO/IEC 12207:2008. 2008. Systems and software engineering - Software life cycle processes. Geneva, Switzerland.

ISO/IEC 14598-1:1999. 1999. Information technology -- Software product evaluation -- Part 1: General overview. http://goo.gl/cuOFkB.

ISO/IEC 15504-X. 2010. Information technology - Process assessment - Parts 1-10. http://goo.gl/lyQ9IR.

ISO/IEC 25000:2014. 2014. Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE. http://goo.gl/WBcuam.

ISO/IEC 25010:2011. 2011. Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models. http://goo.gl/NE3MSc.

ISO/IEC TR 19759:2005. 2007. Software Engineering - Guide to the Software Engineering Body of Knowledge (SWEBOK). http://goo.gl/5vJD3G.

ISO/IEC TR 24774:2010. 2010. Systems and software engineering - Life cycle management - Guidelines for process description. http://goo.gl/3sQgZD.

JENERS (PRICOPE), S. AND LICHTER, H. 2011. A model based integration approach for reference models. In: Product-focused Software Process Improvement: 12th International Conference: PROFES 2011; Torre Canne, Italy, ACM Press, 6–9. http://goo.gl/sYr8i0.

JENERS, S. 2014. MosAIC Model supported Adoption and Assessment of Improvement Concepts. Presented at International Conference on Software Quality - ICSQ 2014, Dallas, Texas, USA.

JENERS, S., CLARKE, P., O'CONNOR, R.V., BUGLIONE, L., AND LEPMETS, M. 2013a. Harmonizing Software Development Processes with Software Development Settings – A Systematic Approach. EuroSPI '13 Proceedings of the 20th European System, Software & Service Process Improvement & Innovation, Dundalk, Ireland, Springer, 167–178. http://goo.gl/J9CdKh.

JENERS, S., CLARKE, P., O'CONNOR, R.V., BUGLIONE, L., AND LEPMETS, M. 2013b. Harnessing Software Development Contexts to inform Software Process Selection Decisions. Software Quality Professional 16, 1. http://goo.gl/yajrkF.

JENERS, S. AND LICHTER, H. 2013. Smart Integration of Process Improvement Reference Models Based on an Automated Comparison Approach. EuroSPI '13 Proceedings of the 20th European System, Software & Service Process Improvement & Innovation, Dundalk, Ireland, Springer, 143–154. http://goo.gl/Wd6Bi3.

JENERS, S., LICHTER, H., AND DRAGOMIR, A. 2012a. Towards an Integration of Multiple Process Improvement Reference Models Based on Automated Concept Extraction. EuroSPI '12 Proceedings of the 20th European System, Software & Service Process Improvement & Innovation, Vienna, Austria, Springer, 205–216. http://goo.gl/nwfaMq.

JENERS, S., LICHTER, H., AND PYATKOVA, E. 2012b. Metric based Comparison of Reference Models based on Similarity. In JDCTA: International Journal of Digital Content Technology and its Applications, AICIT, Gyeongbuk, Korea 6, 21, 50–59. http://goo.gl/rbzTkv.

JENERS, S., LICHTER, H., AND PYATKOVA, E. 2012c. Automated Comparison of Process Improvement Reference Models Based on Similarity Metrics. In: Software Engineering Conference (APSEC) 19th Asia-Pacific, Hong Kong, China, 743–748. http://goo.gl/5aIwlc.

JENERS, S., LICHTER, H., AND ROSENKRANZ, C.G. 2013c. Efficient Adoption and Assessment of Multiple Process Improvement Reference Models. In e-informatica: software engineering journal, Oficyna Wydawn. Politechn., Wroclaw 7, 15–24. http://goo.gl/p3MxDo.

JONES, C. 2007. Development practices for small software applications. Software Productivity Research. http://goo.gl/Qz9zwh.

KALUS, G. AND KUHRMANN, M. 2013. Criteria for software process tailoring: a systematic review. Proceeding ICSSP 2013 Proceedings of the 2013 International Conference on Software and System Process, 171–180. http://goo.gl/qXbSUT.

KAUTZ, K. 1998. Software process improvement in very small enterprises: does it pay off? Software Process: Improvement and Practice 4, 4, 209–226. http://goo.gl/hWKTKq.

KEIL, M., CULE, P.E., LYYTINEN, K., AND SCHMIDT, R.C. 1998. A framework for identifying software project risks. Magazine Communications of the ACM 41, 11, 76–83. http://goo.gl/AqnXMg.

KELEMEN, Z.D. 2013. Process based unification for multi-model software process improvement. Technische Universiteit Eindhoven. http://goo.gl/ahxg2P.

KELLER, K. AND MACK, B. 2013. Maturity Profile Reports. CMMI Institute. http://goo.gl/dbPjdK.

KELLNER, M.I., MADACHY, R.J., AND RAFFO, D.M. 1999. Software process simulation modeling: Why? What. Journal of Systems and Software 46, 91–105. http://goo.gl/hqlK6w.

KENSCHE, D., QUIX, C., CHATTI, M.A., AND JARKE, M. 2007. GeRoMe: A Generic Role Based Metamodel for Model Management. Book Journal on data semantics VIII, 82–117. http://goo.gl/Mtx89F.

KULPA, M.K. 2003. Interpreting the CMMI: a process improvement approach. Auerbach, Boca Raton, FL. http://goo.gl/7M2Hkk.

LAMSWEERDE, A.V. AND LETIER, E. 2003. From Object Orientation to Goal Orientation: A Paradigm Shift for Requirements Engineering. Radical Innovations of Software & System Engineering, Montery'02 Workshop, Venice (Italy), LNCS, Springer, 4–8. http://goo.gl/k8W4AR.

LEMBO, D., LUTZ, C., AND SUNTISRIVARAPORN, B. 2006. Tasks for Ontology Design and Maintenance. Thinking ONtolgiES (TONES). http://goo.gl/lo5qNC.

LEPMETS, M., MCBRIDE, T., AND RAS, E. 2012. Goal alignment in process improvement. Journal of Systems and Software 85, 6, 1440–1452. http://goo.gl/EwKYvJ.

LESZAK, M., PERRY, D.E., AND STOLL, D. 2000. A case study in root cause defect analysis. ACM Press, 428–437. http://goo.gl/E6f4q4.

LEVENSHTEIN, V.I. 1966. Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics-Doklady 10, 8, 707–710. http://goo.gl/jVo2Dg.

LI LIAO, YUZHONG QU, AND LEUNG, H.K.N. 2005. A Software Process Ontology and Its Application. http://goo.gl/75lfVh.

LINDVALL, M., MUTHIG, D., DAGNINO, A., ET AL. 2004. Agile software development in large organizations. Computer 37, 12, 26–34. http://goo.gl/MsXYVK.

LÖWENTHAL, T. 2011. Generierung von web-basierten Prototypen für Geschäftsanwendungen. RWTH Aachen. http://goo.gl/1r6Vwd.

LUDEWIG, J. 2002. Modelle im Software Engineering - eine Einführung und Kritik. Modellierung, 7–22. http://goo.gl/ykS8e5.

LUDEWIG, J. AND LICHTER, H. 2010. Software Engineering: Grundlagen, Menschen, Prozesse, Techniken. dpunkt-Verlag, Heidelberg.

MACCORMACK, A. AND VERGANTI, R. 2003. Managing the Sources of Uncertainty: Matching Process and Context in Software Development. Journal of Product Innovation Management 20, 217–232. http://goo.gl/OnCxJz.

MAGERAMOV, T. 2013. Development of a Management and Visualization Application of Integrated Process Improvement Reference Models. RWTH Aachen. http://goo.gl/IOgf68.

MALZAHN, D. 2009. Assessing - Learning - Improving, an Integrated Approach for Self Assessment and Process Improvement Systems. ICONS '09 Proceedings of the 2009 Fourth International Conference on Systems, 126 –130. http://goo.gl/nKTfok.

MARINO, L. AND MORLEY, J. 2008. Process Improvement in a Multimodel Environment Builds Resilient Organizations. Software Engineering Institute, Carnegie Mellon. http://goo.gl/MpKg5k.

MARNEFFE, M.-C. DE, MACCARTNEY, B., AND MANNING, C.D. 2006. Generating typed dependency parses from phrase structure parses. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC), 449–454. http://goo.gl/xYVTPx.

MINNICH, I. 2002. EIA IS 731 compared to CMMISM-SE/SW. Systems Engineering 5, 1, 62–72. http://goo.gl/FBEJaU.

MOGILENSKY, J. 2009. Pathological Box - Checking:The Dark Side of Process Improvement. Process Enhancement Partners, Inc. http://goo.gl/XRbDK3.

MOORE, J.W. 1999. An integrated collection of software engineering standards. IEEE Software 16, 6, 51–57. http://goo.gl/Hx9oct.

MORA, M., GELMAN, O., O'CONNER, R., ALVAREZ, F., AND MACÍAS-LÚEVANO, J. 2008. A Conceptual Descriptive-Comparative Study of Models and Standards of Processes in SE, SwE, and IT Disciplines Using the Theory of Systems. International Journal of Information Technologies and Systems Approach 1, 2, 57–85. http://goo.gl/nThTkV.

MORGAN, G. 2006. Images of organization. Sage Publications, Thousand Oaks.

MUÑOZ, M., MEJIA, J., CALVO-MANZANO, J.A., CUEVAS, G., AND SAN FELIU, T. 2013. Method to Evaluate Process Performance Focused on Minimizing Resistance to Change. International Journal of Human Capital and Information Technology Professionals 4, 2, 1–15. http://goo.gl/WaEN5Q.

NAU, D.S. 2010. Lecture notes on Introduction to AI. University of Maryland, Department of Computer Science. http://goo.gl/GmZqYU.

NUGUES, P.M. 2006. An introduction to language processing with Perl and Prolog an outline of theories, implementation, and application with special consideration of English, French, and German. Springer, Berlin.

O' MALLEY, B. 2013. Software Quality for Medical Devices and Healthcare. Lero, The Irish Software Engineering Research Centre. http://goo.gl/Tm4G7M.

OZ, E. AND SOSIK, J.J. 2000. Why information systems projects are abandoned: a leadership and communication theory and exploratory study. Journal of Computer Information Systems v41 i1, 66–79. http://goo.gl/SA5S1o.

PARDO, C., PINO, F.J., GARCÍA, F., PIATTINI, M., AND BALDASSARRE, M.T. 2012. An ontology for the harmonization of multiple standards and models. Computer Standards & Interfaces 34, 1, 48–59. http://goo.gl/GQZeOQ.

PAULK, M.C. 1994. A Comparison of ISO 9001 and the Capability Maturity Model for Software. Software Engineering Institute. http://goo.gl/ewQNSI.

PAULK, M.C. 2010. The Impact of Process Discipline on Personal Software Quality and Productivity. ASQ Software Quality Professional, Vol. 12, No. 2, 15–19. http://goo.gl/VEFL9r.

PAULK, M.C. 2014. Adoption of Agile Methods by High Maturity Organizations. Dallas, Texas, USA.

PETERSEN, K. AND WOHLIN, C. 2009. Context in industrial software engineering research. IEEE, 401–404. http://goo.gl/ka14ZT.

PETTERSSON, F., IVARSSON, M., GORSCHEK, T., AND ÖHMAN, P. 2008. A practitioner's guide to light weight software process assessment and improvement planning. Journal of Systems and Software 81, 6, 972–995. http://goo.gl/HmsSb3.

POHL, K. 2011. Requirements engineering fundamentals: a study guide for the Certified Professional for Requirements Engineering exam: foundation level, IREB compliant. Rocky Nook ; Distributed by O'Reilly Media, Santa Barbara, CA, USA. http://goo.gl/jW2DI0.

PROJECT MANAGEMENT INSTITUTE. 2013. A guide to the project management body of knowledge (PMBOK guide). Project Management Institute, Inc, Newtown Square, Pennsylvania. http://goo.gl/qrlwQs.

PYATKOVA, E. 2011. Analysis of Similarity Methods for the Comparison of Reference Models. RWTH Aachen. http://goo.gl/i9uE4c.

RAGAISIS, S., PELDZIUS, S., AND SIMENAS, J. 2010. Mapping CMMI-DEV maturity levels to ISO/IEC 15504 capability profiles. World Scientific and Engineering Academy and Society (WSEAS), 13–18. http://goo.gl/OPqmhB.

RAHM, E. AND BERNSTEIN, P.A. 2001. A Survey of Approaches to Automatic Schema Matching. Journal The VLDB Journal — The International Journal on Very Large Data Bases 10, 4, 334–350. http://goo.gl/FWVuuA.

RAMSHAW, L.A. AND MARCUS, M.P. 1999. Text Chunking using Transformation-Based Learning. Natural Language Processing Using Very Large Corpora, 157–176.

RECTOR, A. 2002. Normalisation of ontology implementations: Towards modularity, re-use, and maintainability. In Proceedings Workshop on Ontologies for Multiagent Systems (OMAS) in conjunction with European Knowledge Acquisition Workshop, Siguenza, Spain, 16. http://goo.gl/jkqenz.

ROUT, T.P. AND TUFFLEY, A. 2007. Harmonizing ISO-IEC 15504 and CMMI. Softw. Process 12, 4, 361–371. http://goo.gl/07jFn4.

SCHMIDT, M., POLOWINSKI, J., JOHANNES, J., AND FERNÁNDEZ, M.A. 2010. An Integrated Facet-Based Library for Arbitrary Software Components. 6th European Conference, ECMFA 2010, Paris, France, Springer Berlin Heidelberg, 261–276. http://goo.gl/gGOp0H.

SEIR. 2013. SEIR. Software Engineering Information Repository from SEI (Software Engineering Institute). http://goo.gl/Flxow3 http://goo.gl/Flxow3.

SHVAIKO, P., EUZENAT, J., SHVAIKO, P., AND EUZENAT, J. 2005. A Survey of Schema-Based Matching Approaches Journal on Data Semantics IV. Journal on Data Semantics IV, 146–171. http://goo.gl/lSWA3p.

SIVIY, J., KIRWAN, P., MARINO, L., AND MORLEY, J. 2008a. The Value of Harmonizing Multiple Improvement Technologies: A Process Improvement Professional's View. Software Engineering Institute, Carnegie Mellon. http://goo.gl/pQVUdG.

SIVIY, J., KIRWAN, P., MARINO, L., AND MORLEY, J. 2008b. Process Architecture in a Multimodel Environment. Software Engineering Institute, Carnegie Mellon. http://goo.gl/RbZ4J0.

SIVIY, J., KIRWAN, P., MARINO, L., AND MORLEY, J. 2008c. Strategic Technology Selection and Classification in Multimodel Environments. Software Engineering Institute, Carnegie Mellon. http://goo.gl/l31RZ4.

SMADJA, F. 1993. Retrieving Collocations from Text: Xtract. Comput. Linguist. 19, 1, 143–177. http://goo.gl/yzg1oN.

SOTO, M. AND MÜNCH, J. 2008. Using model comparison to maintain model-to-standard compliance. Proceedings of the 2008 international workshop on Comparison and versioning of software models, ACM, 35–40. http://goo.gl/flTHbm.

STACHOWIAK, H. 1973. Allgemeine Modelltheorie. Springer-Verlag. http://goo.gl/vmFi4b.

STEPHEN, A. 2013. The ITSM review. News, Reviews and Resources for ITSM Professionals. http://goo.gl/6H07r5.

STOREY, V.C. 1993. Understanding semantic relationships. 455–488. http://goo.gl/WLukgt.

STOTTS, D., WILLIAMS, L., NAGAPPAN, N., BAHETI, P., JEN, D., AND JACKSON, A. 2003. Virtual Teaming: Experiments and Experiences with Distributed Pair Programming. In: F. Maurer and D. Wells, eds., Extreme Programming and Agile Methods - XP/Agile Universe 2003. Springer, 129–141. http://goo.gl/Qm8KL6.

SUBRAMANIAN, G.H., KLEIN, G., JIANG, J.J., AND CHAN, C.-L. 2009. Balancing four factors in system development projects. Communications of the ACM 52, 10, 118. http://goo.gl/5MU9NW.

SUTHERLAND, J. AND SCHWABER, K. 2013. SCRUM Guide. Scrum.org. http://goo.gl/4GkgxF.

TESCH, D., KLOPPENBORG, T.J., AND FROLICK, M.N. 2007. IT project risk factors: the project management professionals perspective. Journal of Computer Information Systems 47, 4, 61. http://goo.gl/0ANZIl.

THE STANDISH GROUP INTERNATIONAL. 2013. CHAOS Manifesto 2013. http://goo.gl/wFwrcP.

THIRY, M., ZOUCAS, A., AND TRISTÃO, L. 2010. Mapping process capability models to support integrated software process assessments. CLEI Electronic Journal 13, 1. http://goo.gl/fOYEYJ.

THOMAS, M. AND MCGARRY, F. 1994. Top-down vs. bottom-up process improvement. IEEE Software 11, 4, 12–13. http://goo.gl/kU0a5y.

TREVELLION, S. 2009. MCIF – a systematic approach to software delivery excellence. IBM Rational.

TVERSKY, A. AND GATI, I. 1978. Studies of similarity. In E. Rosch and B.B. Lloyd, Cognition and Categorization, 79–98. http://goo.gl/y6TgQZ.

UNTERKALMSTEINER, M., GORSCHEK, T., ISLAM, A.K.M.M., CHOW KIAN CHENG, PERMADI, R.B., AND FELDT, R. 2012. Evaluation and Measurement of Software Process Improvement - A Systematic Literature Review. IEEE Transactions on Software Engineering 38, 2, 398–424. http://goo.gl/3S8AR2.

WALLACE, L. AND KEIL, M. 2004. Software project risks and their effect on outcomes. Magazine Communications of the ACM - Human-Computer Etiquette 47, 4, 68–73. http://goo.gl/RhqZ5I.

WANG, Y., KING, G., DORLING, A., AND WICKBERG, H. 1999. A unified framework for the software engineering process system standards and models. Software Engineering Standards, 1999. Proceedings. Fourth IEEE International Symposium and Forum on, 132–141. http://goo.gl/jjyUYn.

VON WANGENHEIM, C.G., HAUCK, J.C.R., SALVIANO, C.F., AND VON WANGENHEIM, A. 2010. Systematic literature review of software process capability/maturity models. Proceedings of International Conference on Software Process Improvement and Capabity dEtermination (SPICE), Pisa, Italy. http://goo.gl/Ck1pNG.

WELTY, C. AND GUARINO, N. 2001. Supporting ontological analysis of taxonomic relationships. Journal Data & Knowledge Engineering - ER2000 39, 1, 51–74. http://goo.gl/A7xMSX.

WIXSON, J.R. 1999. Function analysis and decomposistion using function analysis systems technique. International Council on Systems Engineering Annual Conference (INCOSE '99), 6–10. http://goo.gl/exZRrm.

WOHLIN, C. 2012. Experimentation in software engineering. Springer.

XU, P. AND RAMESH, B. 2007. Software Process Tailoring: An Empirical Investigation. Journal of Management Information Systems 24, 2, 293–328. http://goo.gl/PxTFFU.

YOO, C., YOON, J., LEE, B., ET AL. 2006. A unified model for the implementation of both ISO 9001:2000 and CMMI by ISO-certified organizations. Journal of Systems and Software - Special issue: Selected papers from the 11th Asia Pacific software engineering conference (APSEC 2004) 79, 7, 954–961. http://goo.gl/fMguBM.

ZHOU, X., ZHANG, X., AND HU, X. 2007. The Dragon toolkit developer guide. Data Mining and Bioinformatics Laboratory, Drexel Univ., http://www. dragontoolkit. org/tutorial. pdf. http://goo.gl/xQx9lT.