CMMI-DEV v1.3 Reference Model in ArchiMate

Luís Miguel Gonçalves Valverde
luis.valverde@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

October 2018

Abstract

To verify what are the existing concepts and their relationships in a framework, graphical reference models can be used instead of textual descriptions, giving a different and a clearer idea of how a framework works. Capability Maturity Model Integration (CMMI) is a well-known framework used by organizations to improve their processes. When adopting a framework like CMMI, organizations face many challenges, some of them related to the perceived complexity of the textual representation of CMMI by its users.

In order to reduce the high perceived complexity of CMMI by its users, a graphical reference model for CMMI using ArchiMate as the chosen Enterprise Architecture (EA) modeling language was proposed. The research is focused on the part of CMMI related to the development of both products and services, known as CMMI for Development (CMMI-DEV), in the version 1.3. With ArchiMate as the chosen EA modeling language, a reference model for CMMI-DEV v1.3 was developed following the Design Science Research Methodology (DSRM) with the purpose of reducing the high perceived complexity of this framework by representing its concepts and relationships with graphical concepts and relationships of ArchiMate.

To demonstrate the utility of the proposed reference model for CMMI-DEV v1.3, a field study was conducted in a real organization that was improving their processes using the same version of CMMI-DEV used in this research. The demonstration in the research shows mappings between the EA of the organization and the proposed reference model with the purpose of demonstrating the potential benefits of representing CMMI with an EA.

Finally, to validate both the proposed reference model and the demonstration, well-known techniques to evaluate Design Science (DS) artifacts (mappings, models), as well as interviews with the staff of the organization behind the project of adopting CMMI, were used. The research concludes with some findings and future work on this topic.
Keywords: Reference Models, Capability Maturity Model Integration, Enterprise Architecture, ArchiMate

1. Introduction

Nowadays, organizations are increasingly focusing on the quality and functionality of the software they develop, much of which involves redesigning their software processes to follow the best practices known in the industry so that the software they developed becomes trustworthy, reliable, of a high quality, fit its purpose and is consistently delivered on time to their customers [1].

A software process can be seen as the glue that connects people, tools and equipments, and procedures and methods in a consistent way through a set of interrelated activities that, together, interact to develop and maintain software and the associated work products [1, 2]. The Software Engineering Institute (SEI) that is an American research, development and training center involved in computer software and network security and other quality experts believe that there is a close relationship between the quality of the delivered software and the quality and maturity of the software processes [1]. To achieve mature processes with improved quality and effectiveness, the steps that organizations take are not so intuitive as they can do it without a certain type of guidance, therefore there was a need to create initiatives that lead to a focus on software processes and on ways to improve them [3].

Software Process Improvement (SPI) initiatives define and measure best practices and processes for improving the existing processes in organizations, intending to help them develop higher quality software and products and achieve their business goals more efficiently, where the business goals can be: faster delivery of software and products to the market on time and budget; improved customer satisfaction; improved quality of software and products; and cost reduction of development. In other words, SPI helps organizations to work smarter, as they build software and products, better, faster and cheaper than the competitors, providing a faster Return on Investment (ROI) [1].

There are some international standards, frameworks, and models that define different programs for SPI, these are a set of best practices that are aligned with the business goals and play a key role in helping organizations to achieve their strategic goals. From all the models for SPI, the one that is more related to software development (practices for processes in software and systems engineering) is CMMI. CMMI contains a set of best practices from several areas, each of them to achieve a given purpose, that can include tools, methods, materials, and people. Organizations follow these practices to improve their software processes and meet their business needs more efficiently, as well as allowing them to do continuous improvement, adapt to technological innovations, and to prevent defects, in order to them to face future challenges [1, 2].

Currently, the CMMI framework addresses three constellations, that are known as collections of CMMI components for a specific area of interest. The focus of this research is in the development constellation also known as CMMI-DEV in the version 1.3 that is related to the development of both products and services [2].

Due to the high complexity of CMMI-DEV in version 1.3 many organizations struggle to understand the model as they get lost in the various concepts and relationships in the model. To reduce the perceived complexity of the model, we propose to represent CMMI-DEV v1.3 with an EA using for that matter, the ArchiMate modeling language to represent both concepts and relationships of the framework and test if the reading of model in ArchiMate can be easier for the model users as well as analyze potential benefits of representing it with an EA.

2. Research Problem

The main benefits of adopting CMMI for process improvement are the reduction of the overall costs, project schedule (improvements in schedule predictability and reductions in the time required to do the work), quality improvement (reductions in the number of defects), customer satisfaction and ROI. These benefits have been proven by many case studies of organizations that adopted CMMI from around the world and from different areas [4]. Despite this, only a little fraction of the software developing organizations adopt CMMI, so it is important to find why organizations do not adopt CMMI.

The authors from [5] studied why organizations do not adopt CMMI by collecting and analyzing two months of sales data from an Australian company that sells CMMI appraisals and improvement services. Their findings show that most organizations do not adopt CMMI because the CMMI program is expensive, requires time that many organizations do not have, organizations do not understand the benefits of using it and many organizations think they are too small to adopt it [this being a direct influence from not understanding the benefits of CMMI and resources constraints (budget and time)].

Two years later the authors from [6] replicated the previous study in another country, Malaysia, by using data collected from three consulting companies that sell a CMMI Level 2 program subsidized by the Malaysian government. This study supports the study conducted by [5] and identifies the same adoption problems as well as the problem of organizations having other priorities than process improvement.

Another study [7], now related to organizations who have already adopted CMMI, was done through a study conducted in China with the purpose of investigating the adoption of CMMI. The authors inquired most of the organizations who have been rated in a certain CMMI assessment to find the reasons, success factors, benefits and problems in the adoption of CMMI. Through the survey's data, they identified the following problems: the organizations think that CMMI is an overcomplex and dogmatic process, the costs are high when adopting it and that there is a lack of automated supporting tools for CMMI.

Some of the problems previously mentioned regarding the low adoption and the adoption of CMMI are directly related to its complexity and difficulty to be understood. The existing textual reference model of CMMI-DEV v1.3 [2] is ambiguous, has a lot of technical definitions and extensive text, which make difficult for users to understand, implement and accept it [8]. Users tend to get lost as there are a total of twenty-two process areas and near two hundred practices in the model with various relationships between them, increasing the effort of users in reading this representation of the reference model. This, allied to the fact that CMMI tells what to do and not how to do it for process improvement increases its complexity.

Summarizing, the problem that this research will tackle is the high perceived complexity of CMMI-DEV v1.3 by its users. To tackle this problem we are going to propose a different way to represent the CMMI-DEV v1.3 model, by using EA models with graphical elements of ArchiMate which usually are more appellative to people and easier to understand.

3. Research Methodology

In this research we are going to try to solve our research problem through the development of models and instantiations of these models (new artifacts) fitting in the Design Science paradigm. To guide a DS research, we propose to use the DSRM. The DSRM guides our work through the development and evaluation of our various Information Systems (IS) artifacts.

The DSRM is an iterative methodology that incorporates principles, practices, and procedures to carry out a DS research in IS area by creating and validating artifacts to address a research problem. These artifacts are broadly defined as constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), and instantiations (implemented and prototype systems) [9, 10].

In our research, the artifacts that we are going to create and evaluate are going to be models and constructs, by models we are referring to the reference model of CMMI and by constructs the mapping we chose between the CMMI and ArchiMate.

The DSRM is divided into 6 phases: Problem identification and motivation, Define the objectives for a solution, Design, and development, Demonstration, Evaluation, and Communication.

The DSRM by being an iterative process lets us iterate many times through various phases and in each iteration obtain frequent and valuable feedback for the design process and incremental improvement of it.

The structure of this research is strongly influenced by the DSRM, following the various phases of the methodology. Chapter 1 starts by introducing the research, giving detail to the research theme, the research problem, and motivation behind the research work (first phase of DSRM), chosen methodology and structure of the research. Chapter 4 does not follow a specific phase of the DSRM but is essential to our a research, as introduces the fundamental concepts and other author's researches regarding the scope of our own research. Chapter 5 follows the second and third phase of DSRM by defining the objectives of the proposed solution and the proposed solution. Chapter 6 follows the fourth phase of DSRM by demonstrating the proposed solution in practice. Chapter 7 follows the fifth phase of DSRM by evaluating the utility of the proposed solution. The final Chapter 8 have the conclusions and contributions of this research.

4. Related Work

In this chapter, we start by giving a theoretical background of the main topics related to our research and introduce some researches that are relevant to support our proposed solution. For the theoretical background, we start by introducing CMMI with a main focus on the Development constellation (CMMI-DEV) in the version 1.3, next we introduce the EA area and the modeling language used in this research, ArchiMate. For the related work, we introduce proposals of ontologies for CMMI and researches that use the ArchiMate language to model other frameworks that have similar concepts to CMMI, like COBIT 5, ITIL and some ISO's.

4.1. Capabilty Maturity Model Integration

CMMI was developed by a group of expert professionals from industry, government, and the SEI at Carnegie Mellon University and it is used to guide organizations to improve their processes and, consequently, improve software quality, time and costs of development and productivity by describing an evolutionary improvement path from ad hoc, immature processes to disciplined, mature processes with improved quality and effectiveness, telling organizations what to do to but not how to do it to achieve the desired business goals [11, 12].

In our research, we are going to focus on the Development constellation also known as CMMI-DEV, that is used by organizations from different industries to cover activities for developing both products and services. CMMI-DEV is composed of practices that cover project management, process management, systems engineering, hardware engineering, software engineering, and other supporting processes used in the development and maintenance of projects. The version 1.3 is constituted by twenty-two process areas divided into four categories: Process Management, Engineering, Project Management and Support [11, 12].

4.2. Enterprise Architecture

Currently, to manage the complexity of any large organization or system, an architecture is needed. Architecture can be described as the properties and concepts that are fundamental to a system in their environment, incorporated into its elements, relationships and in the principles required for its analysis, design and evolution [13].

The use of architecture in an organization gave form to the EA term which is defined by Lankhorst [13] as *"a coherent whole of principles, methods, and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems, and infrastructure"*.

Nowadays, business performance increasingly depends on a balanced and integrated EA, involving stakeholders, their competencies, organization structures, business processes, Information Technologies, finances, products, and services, as well as its environment. EA is seen as a holistic view of the representation of an organization and helps design the various layers of the EA in an organization [13, 14].

One of the advantages of using EA models to describe these types of IS standards, frameworks and models is the fact that EA models are more readable and understandable than textual descriptions, lowering the user's perceived complexity and therefore facilitates the learning of this type of frameworks.

4.3. ArchiMate

ArchiMate is an EA modeling language developed by The Open Group, with focus, mainly in enterprise modeling, allowing to describe, analyze and visualize business architectures, using for this purpose graphical concepts for the entities and relations. ArchiMate provides uniform representations of EAs by the form of models, that contain well-defined relationships allowing to connect concepts from different domains of the business architectures. In order to do this, it defines several layers in the EA, relating and distinguishing each of them with a service-oriented approach, that is, the layers expose their functionality as a form of services to the layers above [15].

ArchiMate also defined a core set of generic relationships that can be used to connect the generic concepts and give a meaning to the way we look at the models. These relationships are *"overload"*, which means that the exact meaning of the connection between two concepts differs, depending on the source and destination concept [15].

4.4. CMMI Ontologies

There is no proposal on how to model CMMI in ArchiMate, but there are authors that propose other CMMI ontologies and metamodels using different modeling languages.

In the paper [16], the authors propose a CMMI-DEV v1.3 ontology based on Web Ontology Language (OWL), a primary language for the Semantic Web. The authors followed the same approach used in this language, first they formalize an ontology for the CMMI-DEV v1.3 that captures the main concepts and then they use a generic OWL reasoner to check the consistency of the representation of the CMMI-DEV 1.3 ontology to derive the classification of the level of maturity of the organization's development process. This will allow to determine the maturity levels of organizations through their data on the practices performed.

In the paper [17], following the paradigm Model Driven Development (MDD), the authors proposed a tool that supports the automatic generation of a language that can be used to specify practices of process areas. This generation is performed through a CMMI metamodel in Unified Modeling Language (UML) that they propose as well.

These researches were an important contribution to our work, as they will be used as a starting point for developing the proposed CMMI-DEV v1.3 metamodel in ArchiMate that contains the main concepts of CMMI-DEV v1.3 and their relationships.

4.5. ArchiMate and Information Systems Frameworks

The use of the ArchiMate modeling language to model frameworks related to the IS area and to enable the mapping with a standard-based EA representation it is well documented. We will highlight some of the researches conducted on this topic:

Lourinho [18] proposes in ArchiMate a metamodel for the ISO 27001 standard extended with the concepts of ISO TS 33052 and 33072, with the purpose of reducing the perceived complexity of implementing these IT frameworks and maps this metamodel with COBIT 5 metamodel proposed by Almeida [19] to show a complementary way of integrating the two. The same COBIT 5 metamodel, is used in another research by Percheiro [20], that proposes a metamodel in ArchiMate on how to integrate COBIT 5 with ITIL.

In other researchers, Vicente [21, 22] proposes in ArchiMate a business motivation model for ITIL and a business-specific architecture using the principles of ITIL and the EA approach.

And finally, Silva [23] proposes in ArchiMate a model for TIPA, a framework specifically used for assessing maturity for those who use ITIL and demonstrates how the model in ArchiMate allows alignment between service management and the organization's concepts and artifacts in a standardized way.

These researches were an important contribution to our work, as they will be used as a starting point when mapping ArchiMate with CMMI, these frameworks have concepts that have similar meanings to the CMMI concepts, and the ArchiMate concepts that these authors chose to represent them can be used in our research.

5. Proposal

This chapter corresponds to the "Define the objectives for a solution" and "Design and development" phases of DSRM [9], where we explain our research proposal to solve the research problem as well as define the main objectives that the proposed solution must satisfy in order to address the research problem.

The purpose of the CMMI-DEV v1.3 reference model is to facilitate the learning of this framework, as well as understand what are the main components and their relationships. To solve the problem previously identified, we propose a metamodel of CMMI-DEV v1.3 developed in ArchiMate, containing the main components and relationships between them, and further ahead a model more detailed that we named CMMI-DEV v1.3 Reference

Table 1: Mapping CMMI concepts with ArchiMate

CMMI	ArchiMate	Justification [2, 15]
Process Area Category	Grouping	A Process Area Category in CMMI represents a group of Process Areas from the same area of interest and can be represented by the Grouping concept which in turn can represent an element that composes concepts that belong together based on some common characteristic.
Process Area	Grouping	A Process Area in CMMI is defined as a cluster of related practices and can be represented by the Grouping concept which in turn can represent an element that composes concepts that belong together based on some common characteristic.
Purpose	Goal	A Purpose statement in CMMI describes the purpose of the process area and can be represented by the Goal concept which in turn can represent a high-level statement of intent, direction, or desired. In this case, it is due to the purpose of ArchiMate represent the intention of satisfying the process area.
Specific Goal	Goal	A Specific Goal in CMMI describes the unique characteristics that must be present to satisfy the process area and can be represented by the Goal concept which in turn can represent a high-level statement of intent, direction, or desired. In this case, it is due to the fact that the goals represent the desired characteristic to implement a process area.
Generic Goal	Goal	A Generic Goal in CMMI is called "generic" because the same goal statement applies to multiple process areas, and can be represented by the Goal concept for the same reason as in the Specific Goal.
Specific Practice	Business Process	A Specific Practice in CMMI is the description of an activity that is considered important in achieving the associated specific goal and can be represented by the Business Process concept which in turn can represent a human activity.
Generic Practice	Business Process	A Generic Practice is called "generic" because the same practice applies to multiple process areas and can be represented by the Business Process concept for the same reason as in the Specific Practice.
Subpractice	Business Process	A Subpractice in CMMI is defined as a detailed description that provides guidance for interpreting and implementing a specific or generic practice and can be represented by the Business Process concept for the same reason as in the Specific and Generic Practice.
Example Work Products	Business Object	The Example Work Products in CMMI defines section lists of sample outputs from a specific practice and can be represented by the Business Object concept which in turn can represent information produced and consumed by an activity.
Generic Practice Elaborations	Deliverable	Generic Practice Elaborations appear after generic practices to provide guidance on how the generic practices can be applied uniquely to a process area and can be represented by the Deliverable concept, which in turn can represent a support concept.

Model. The metamodel has the main concepts of the CMMI-DEV v1.3 and their relationships, giving an overview with a focus on the metaclasses that represent them. The reference model is a more detailed model that has all the instantiations of the metaclasses identified in the metamodel and represent the full CMMI-DEV v1.3 framework using several views.

To create our models in ArchiMate, we first chose which concepts of ArchiMate could represent the CMMI concepts. In order to do this, we did the mapping shown in Table 1 based on the previous studies in using ArchiMate with IS frameworks (Section 4.5). Then we identify the main relationships between the CMMI concepts and chose which ArchiMate relationships to represent them based on the textual reference model of CMMI [2] and the previous studies of CMMI ontologies (Section 4.4).

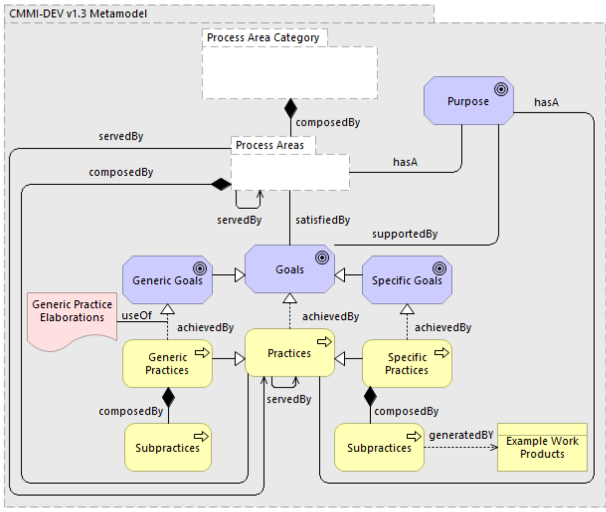

Figure 1: CMMI-DEV v1.3 Metamodel in ArchiMate

For the relationships, we chose the composition relationship of ArchiMate to represent that a concept of CMMI is constituted by others CMMI concepts. We use the serving relationship when a concept of CMMI provides something to another concept of CMMI, the realization relationship is used when a concept of CMMI helps to achieve another CMMI concept, the access relationship is used when referring that a concept of CMMI accesses (create) another CMMI concept and finally, for the association relationship, we use it when there are CMMI concepts that have some type of association that cannot be represented by other ArchiMate relationship.

Based on the ArchiMate concepts and relationships that we chose to represent CMMI-DEV v1.3, we proposed a metamodel, as shown in Fig. 1 which can be read as follows: a process area category is composed of a set of process areas, each process area has a purpose and each process area is composed of practices (specific and generic) that also have a purpose. To satisfy a process area all the goals (specific and generic) have to be achieved or satisfied, and this can be done by performing all the practices (specific and generic) related to each goal, the achievement of all the goals also support the purpose of the process area. To achieve the generic goals in a particular process area the generic practice elaborations are used to provide guidance on how the generic practices can be applied. The practices are composed by subpractices, in the case of the subpractices of specific practices to verify if it is being implemented, there are example work products that are sample outputs of these subpractices. Each process area can be supported by other process areas as well as each practice can be supported by other practices and process areas.

The reference model shows all the instantiations of the metaclasses identified in the metamodel, this model shows all the concepts of CMMI-DEV v1.3 and their relationships in a more detailed way. In Fig. 2 we have a part of the reference model. As we can see in this part of the model, the first level of abstraction contains the process areas categories all of them except Engineering are divided by basic and advanced relationships and each of them contains a set of process areas [2].

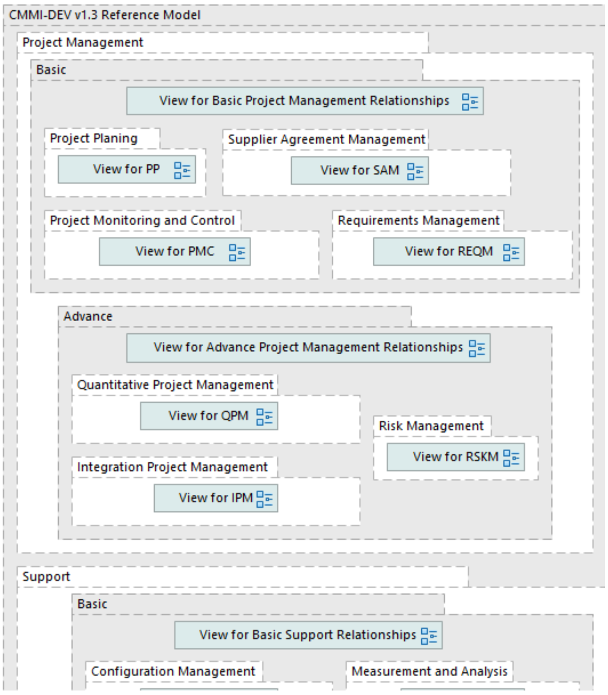

Figure 2: Part of CMMI-DEV v1.3 Reference Model

To navigate to other parts of the reference model we have the views concepts, that are used as links to other parts of the model and if clicked on, will give a more detailed model of the concept they are referring to. For instance to navigate to a process area we click on the view related to that area, for example, , if we click on the “View for REQM” it will show a detailed model of the process area Requirements Management (REQM) as it is shown in Fig. 3. This part of the model contains all of the concepts of the process area, the purpose of the area, all the practices that compose the area, divided in specific and generic as well as all the high-level relationships with related process areas. From this model we can see that the process area Risk Management provides the “Identification and Analysis of Risks” to this process area. For each type of practices, we a have different view, to give in more detail the practices that are necessary to achieve the goals and satisfy the process area.

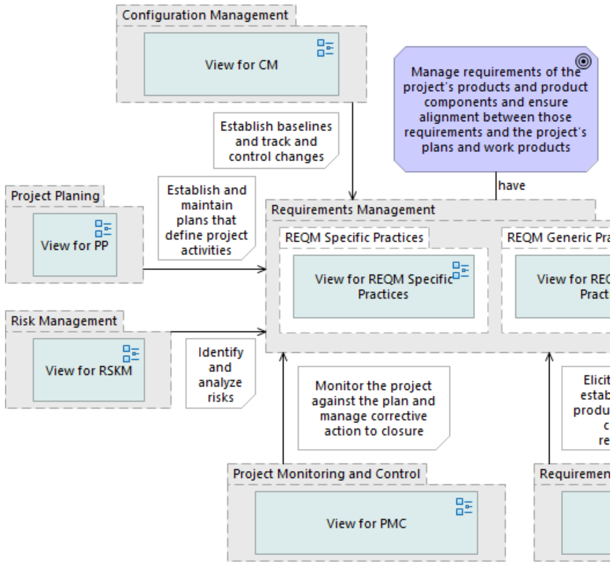

Figure 3: Part of REQM Process Area Model

For the specific practices of this process area, we have the “View for REQM Specific Practices” that allows us to navigate to a detailed model of the specific practices of this process area. This view is shown in Fig. 4 as well as the view for the specific practice 1.2 ”Obtain Commitment to Requirements”. In the part of Fig. 4 containing all the specific practices we can see that the process area REQM is satisfied by achieving the specific goal ”Manage Requirements”, that can be accomplished by performing the five specific practices that compose the area. Achieving this goal also supports the purpose of the REQM area. By clicking in the ”View for REQM SP 1.2” we navigate to the detailed model of the specific practice ”Obtain Commitment to Requirements” shown in Fig. 4, we can see in this model what is the goal that this specific practice helps achieve, the subpractices that compose this specific practice, the example of work products of each subpractice and the process areas that support this specific practice. So, we can see in the model for instance, that the subpractice ”Assess the Impact of requirements on Existing Commitments” has as output the ”Requirements Impact Assessments” as well as see that the process area Project Monitoring and Control provides the ”Monitor of Commitments” to this specific practice.

The full CMMI-DEV v1.3 reference model has twenty-two process areas divided by the four process area categories. Then, we have a view for each process area, for each type of practices in a

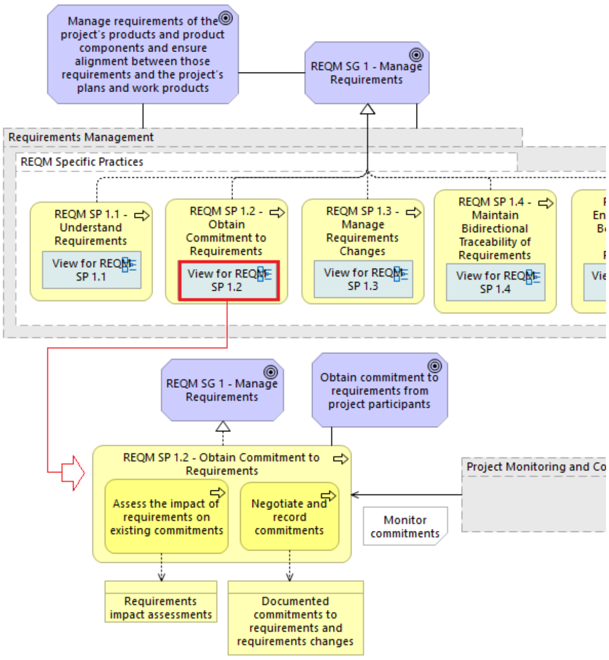

Figure 4: REQM Specific Practices and Specific Practice 1.2

process area and for each practice in the process areas. The model contains as well for each process area views for the generic goals and practices and for each generic practice. In total the proposed reference model has more than 450 views.

6. Demonstration

This chapter corresponds to the "Demonstration" phase of DSRM [9], where we demonstrate the utility of our proposed solution in solving one or more instances of the research problem. For this matter, we did a field study in a Portuguese organization and did the demonstration with the mappings between their EA and the proposed reference model for CMMI-DEV v1.3 to show the benefits of representing CMMI-DEV v1.3 with an EA.

The field study was conducted in one of the biggest technological Portuguese organizations. One of their departments is adopting CMMI, more precisely, trying to achieve Level 3 of maturity to became CMMI-DEV v1.3 Level 3 certified\rated. To achieve this level of maturity they must satisfy all specific goals related to the process areas that compose this level of maturity and levels above this. In total, they must satisfy the specific and generic goals of twenty process areas from the various category. To satisfy each specific goal they must give response to all specific practices related to each specific goal. When we say give response to a specific practice, we are saying that, in the organization, there has to be artifacts (activities, procedures, etc) and affirmations (from interviews) that meet the specific practices.

First, we start by modeling the AS-IS state of the organization. This state has the processes, documents, and tools used in the organization which are part of the scope of CMMI project, that is, the state of the EA before the adoption of CMMI. Then, we did mappings between the specific practices of each process area from CMMI-DEV v1.3 reference model and the organization AS-IS state. These mappings show the parts of the AS-IS state that are giving response to the specific practices of each process area from the reference model. We can achieve these mappings by reading both, CMMI-DEV v1.3 process areas sections in [2] and the organization documentation.

To exemplify one of these mappings we are going to show in Figure 5 a model with the parts of the AS-IS state from the organization software development area that are giving response to the specific practices of the Verification (VER) process area of CMMI-DEV v1.3. At the top of this model we have all the specific practices that compose the process area VER and in the rest of the model, the AS-IS state of the organization related to software development, the process, procedures, templates, and tools used. We then have the relationships we found between the VER process area and this state of the organization EA, that identify the activities and documents that give response to the VER specific practices. For instance, we can see in this mapping that three of the specific practices related with peer reviewing of the process area VER (SP 2.1-2.3) are not mapped with any part of the AS-IS state of the organization as well as see that the specific practices VER SP 1.1, 1.2, 3.1 and 3.2 are being satisfied with activities of the Definition phase of the Iterative Software Development process.

We use the association relationship to indicate that an element of the EA gives response to a specific practice. These relationships have different colors to facilitate the reading of this mapping, for each specific practice we have a different color to represent the relationships associated with it, for instance, we use the green color to specify the elements of the EA that give response to the specific practice VER SP 1.2.

To see in detail what are the activities in the Definition phase of the Iterative Software Development process that give response to the specific practices VER SP 1.1, 1.2, 3.1 and 3.2, the model contains the view elements, that lets us navigate to a more specific model for each element of the model specifying in more detail the relationships between the element and the specific practices.

With this kind of mappings, we can represent in a different way a gap analysis, where we compare the actual state of the organization with the desired state in terms of an EA being CMMI compliant, identifying the strengths and improvement po-

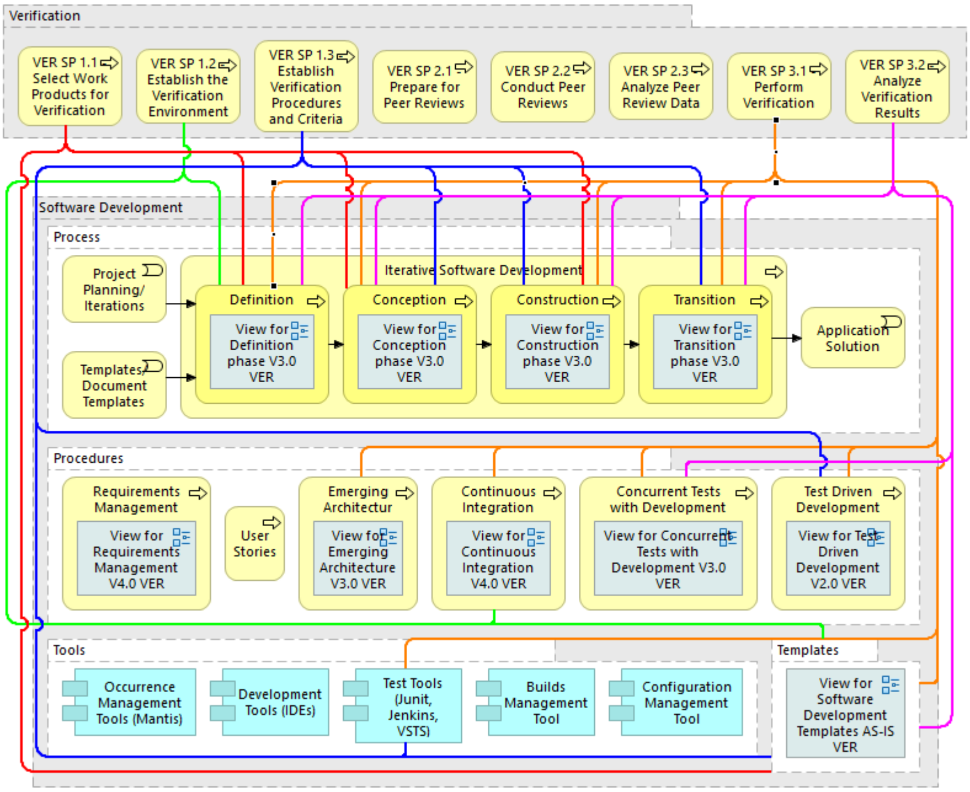

Figure 5: Mapping of VER process area with AS-IS state

tentials regarding CMMI. In other words, we identify what are the specific practices from CMMI-DEV v1.3 that are not being satisfied in the current state of the organization (AS-IS), for the organization to know what are the practices they should focus on when developing and changing their EA to be CMMI compliant. These mappings also show what parts of the existing EA can be reused when developing the new EA of the organization compliant with CMMI-DEV v1.3, that is the TO-BE state.

For the other part of our demonstration, we used the same organization, but now they are already in a stage where they changed their EA to be CMMI compliant, that is, they change their EA to give response to all the specific practices of all the process areas from Level 3 of maturity. Using the same approach as we did with the mappings between the AS-IS state and process areas of CMMI-DEV v1.3, we started by modeling the TO-BE state of the organization and then mapped with the specific practices of CMMI-DEV v1.3 process areas from the reference model. To compare the previously mapping we used again the VER process area and software development area of the organization.

With this kind of mappings, we can check if the new EA was developed to be compliant with CMMI-DEV v1.3. To see if the EA was correctly developed, all the specific practices of the process areas of CMMI-DEV v1.3 have to be accomplished (mapped with the EA).

Having both EA states of the organization, we then color them to emphasize what were the changes in the organization to be compliant with CMMI-DEV v1.3. We use the colors Red, green and blue. Red for parts of the EA that have been removed (presented in AS-IS but not in TO-BE), green for parts of the EA that are new (presented in TO-BE but not in AS-IS) and blue for parts of the EA that were reused (presented in AS-IS and TO-BE).

By coloring both the AS-IS and TO-BE state of the organization we show them a different way to see the changes they made to their EA while adopting CMMI-DEV v1.3. Giving an idea of what were the changes and by locking to the mappings between the TO-BE state and the process areas of the reference model the organization can see the purpose of the change, that is, what specific practices do the new parts of EA give response to. This can also be useful for the organization for futures migrations of CMMI, as they can check what were the changes and why the changes to be compliant with CMMI-DEV v1.3 and see this as a blueprint for the adoption of CMMI-DEV v1.3, and further ahead use this knowledge for future migrations of CMMI.

7. Evaluation

This chapter corresponds to the "Evaluation" phase of DSRM [9], where we evaluate the utility of our proposed solution to tackle the defined research problem of the high perceived complexity of CMMI-DEV v1.3. With the use of the demonstration in a real organization, we did interviews with open discussions with relevant stakeholders in the organization to validate both the utility of the artifact and the demonstration itself.

Interviews done to relevant stakeholders can be used as suitable data generation method to validate and get feedback about the utility of the proposed solution and demonstration. From the interviews, we can get depth information about the utility of our models from the various stakeholders interested in this theme (CMMI professionals, professors and staff from organizations that are adopting or adopted CMMI). This happens because there's more interaction with the stakeholders and some topics arise during the interviews as well ask follow-up questions [24].

Therefore to validate our proposal, we interviewed five IT people from the organization we use to demonstrate our proposed solution, these people were part of the team responsible to adopt/implement CMMI-DEV v1.3 Level 3 of maturity in one of the departments in the organization and had formation in this version of CMMI, so they are familiar with CMMI and its terms.

Before the interviews, we presented our work to them, showing them our proposed CMMI-DEV v1.3 reference model and demonstrating to them the utility of representing CMMI with an EA, by showing the mappings with did between the reference model and their EA before (AS-IS) and after the adoption of CMMI (TO-BE). After the presentation, we ask them some questions.

Unanimous the opinions of the interviewees were positive. For them, the new CMMI-DEV v1.3 reference model is a good way to represent this framework to users in general. They also, stressed out that the proposed representation of CMMI with an EA can be very useful because of the mappings that can be done between this representation and the EA of an organization. Mappings, that can represent the steps of an organization when adopting CMMI (gap analysis, process and practices mapping, etc) and be used for formations in the organization and as CMMI adoption history that can be used in future migrations.

8. Conclusions

This research was developed with the purpose of solving the problem of the high perceived complexity that users have when reading the textual representation of the CMMI model, with a focus on the version 1.3 of CMMI-DEV. There are already a few studies that approached some problems related to other IS frameworks using EA and ArchiMate to tackle them but there is little research about EA and ArchiMate with CMMI.

Thus, to solve this problem, we chose to use an EA approach and develop a reference model using the modeling language ArchiMate for the CMMI-DEV v1.3. With the visual representation of this reference model and based on the interviews we did, we believe that we can lower the user's perceived complexity of CMMI and therefore contribute to turning the CMMI framework easier to use, allowing users to read and understand the CMMI framework more easily and in a more interactive way.

Also, we can take advantage of using the EA approach of representing CMMI with the ArchiMate modeling language, for instance, we can represent the assessment in an organization with models, showing them a different way to see a gap analysis by mapping CMMI practices of the reference model with the present state of an organization (AS-IS) and show what parts of the organization EA give response to CMMI and what practices are not satisfied.

We also can do mappings between the CMMI-DEV v1.3 reference model and the EA of an organization who already adopted CMMI to verify if their EA is compliant with this version of CMMI, that is, all the specific practices of CMMI are being mapped with the EA of the organization.

Having both mappings, we can colored both states of an organization to identify what where the changes in the EA of the organization after adopting CMMI (AS-IS vs TO-BE), underlying the changes they had to make to be CMMI compliant and with the mappings, identify the causes of the changes (practices that are now satisfied by the referred change).

References

[1] Gerard O'Regan. *Concise Guide to Software Engineering From Fundamentals to Application Methods*. Springer International Publishing, 2017.

[2] CMMI Product Team. CMMI for Development, Version 1.3. Technical report, Software Engineering Institute, 2010.

[3] Robert Fantina. *Practical Software Process Improvement*. Artech House, Inc., Norwood, Massachusetts, 2005.

[4] Dennis R. Goldenson and Diane L. Gibson. Demonstrating the impact and benefits of CMMI: an update and preliminary results. Technical report, Software Engineering Institute, 2003.

[5] Mark Staples, Mahmood Niazi, Ross Jeffery, Alan Abrahams, Paul Byatt, and Russell Mur-

phy. An exploratory study of why organizations do not adopt CMMI. *Journal of Systems and Software*, 80(6):883–895, 2007.

[6] Nazrina Khurshid, Paul Bannerman, and Mark Staples. Overcoming the First Hurdle: Why Organizations Do Not Adopt CMMI. *International Conference on Software Process, Trustworthy Software Development Processes*, 5543:38–49, 2009.

[7] Zhanchun Wu, David Christensen, Mingshu Li, and Qing Wang. A Survey of CMM / CMMI Implementation in China. *Software Process Workshop*, pages 507 – 520, 2005.

[8] Zahoor Ul Islam and XIianzhong Zhou. *Software Process Improvement Framework for Software Outsourcing Based On CMMI*. PhD thesis, University of Gothenburg Chalmers University of Technology, 2011.

[9] Alan R Hevner, Salvatore T March, Jinsoo Park, and S. Ram. Design Science in Information Systems Research. *MIS Quarterly*, 28(1):75–105, 2004.

[10] Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3):45–77, 2007.

[11] Mary Beth Chrissis, Mike Konrad, and Sandy Shrum. *CMMI for development: guidelines for process integration and product improvement*. Addison-Wesley Professional, 2011.

[12] Mukund Chaudhary and Abhishek Chopra. *CMMI for Development Implementation Guide*. Apress, 2017.

[13] Marc Lankhorst. *Enterprise Architecture at Work Modelling, Communication and Analysis*. Berlin, Heidelberg, fouth edition, 2017.

[14] Danny Greefhorst and Erik Proper. *Architecture Principles The Cornerstones of Enterprise Architecture*, volume 6. Springer, Berlin, Heidelberg, 2011.

[15] The Open Group. ArchiMate® 3.0.1 Specification, 2017.

[16] Gokhan Halit Soydan and Mieczyslaw M. Kokar. A Partial Formalization of the CMMI-DEV—A Capability Maturity Model for Development. *Journal of Software Engineering and Applications*, 5(10):777–788, 2012.

[17] David Musat, Víctor Castaño, Jose A. Calvo-Manzano, and Juan Garbajosa. MATURE: A model driven based tool to automatically generate a language that supports CMMI process areas specification. *Communications in Computer and Information Science*, pages 48–59, 2010.

[18] Renato Lourinho, Rafael Almeida, Miguel Mira da Silva, Pedro Pinto, and Beatrix Barafort. Mapping of Enterprise Governance of IT Practices Metamodels. *European, Mediterranean, and Middle Eastern Conference on Information Systems*, pages 492–505, 2017.

[19] R. Almeida, P. L. Pinto, and M. M. da Silva. Using ArchiMate to integrate COBIT 5 and COSO metamodels. *Proceedings of the 13th European Mediterranean and Middle Eastern Conference on Information Systems*, 2016.

[20] B Percheiro, Rafael Almeida, Pedro Linares Pinto, and Miguel Mira. Towards Conceptual Meta-Modeling of ITIL and COBIT 5. *European, Mediterranean, and Middle Eastern Conference on Information Systems*, pages 478–491, 2017.

[21] Marco Vicente, Nelson Gama, and Miguel Mira da Silva. Using archimate to represent ITIL metamodel. *Proceedings - 2013 IEEE International Conference on Business Informatics, IEEE CBI 2013*, pages 270–275, 2013.

[22] Marco Vicente, Nelson Gama, and Miguel Mira da Silva. Modeling ITIL Business Motivation Model in ArchiMate. *International Conference on Industrial Engineering and Service Science*, pages 86–99, 2013.

[23] Nuno Silva, Miguel Mira da Silva, Béatrix Barafort, Marco Vicente, and Pedro Sousa. Using ArchiMate to model a process assessment framework. *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 1189–1194, 2015.

[24] Daniel W. Turner. The Qualitative Report Qualitative Interview Design: A Practical Guide for Novice Investigators. *The Qualitative Report*, 15(3):754–760, 2010.