A Reference Model for Migrating from CMMI-DEV v1.3 to CMMI v2.0

Mariana Silva
mariana.s.silva@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

October 2019

Abstract

In today's high-technology business environment, the success of an organization is highly influenced by the functionality and quality of the software they use and develop. The challenge is to deliver reliable software on time and on budget. CMMI helps companies improve their software development processes and, although the benefits are clear, the CMMI textual reference models are complicated. With the new version of CMMI, new concepts and relationships were introduced, thus we propose a CMMI v2.0 Reference Model in ArchiMate to facilitate the migration for companies that are already CMMI-DEV v1.3 accredited. To guide our work we used the Design Science Research Methodology and the utility of the model is demonstrated in a real-world organization that is CMMI-DEV v1.3 accredited and needs to migrate to CMMI v2.0. The demonstration is based on CMMI models in ArchiMate, together with models of the AS-IS and TO-BE enterprise architecture of the organization. To validate the proposed reference model and the demonstration, we used questionnaires and interviews to CMMI experts and practitioners, well as well-known techniques to evaluate Design Science artifacts.

Keywords: Reference Model, Capability Maturity Model Integration, Enterprise Architecture, ArchiMate.

1. Introduction

Nowadays, in the current business practice, an integrated approach to business and Information Technology (IT) is indispensable. The propagation of IT is an enterprise reality and the success of the business is vastly influenced by the functionality and quality of the software companies use and develop.

To follow the best practices known in the industry, organizations increasingly focus on redesigning their software processes aiming at a more reliable software that fits its purpose and is delivered to customers on time and on budget [18].

Software Process Improvement (SPI) initiatives make processes the focal point, which is important since many of the problems in software development companies are caused by faulty processes rather than by people [18].

SPI has become an indispensable tool for software engineers and managers to accomplish their goals since it provides a Return on Investment (ROI) to the organization. It helps software companies deliver the agreed software on time and on budget and improves the quality of the delivered software, while reducing the cost of development and improving customer satisfaction [18].

To support SPI, there are several standards and frameworks available. In this research, we are using Capability Maturity Model Integration (CMMI).

CMMI is an internationally recognized model, used worldwide by thousands of organizations. According to CMMI Institute, this framework consists of ”a set of best practices that enable businesses to improve performance of their key business processes” [7].

Benefits of this framework include improvements in several categories such as cost, schedule, productivity, quality, customer satisfaction and ROI [7, 9].

Although CMMI has clear benefits, studies show that the program is expensive and takes a lot of time and resources to implement [11, 14, 24]. One reason for it is that CMMI models are complicated. The existing textual reference models contain very extensive text, various technical concepts, and numerous relationships between different CMMI practices. Additionally, there are many different concepts in the two most recent versions of CMMI.

With the release of CMMI version 2.0 (CMMI v2.0), if a company that is CMMI for Development version 1.3 (CMMI-DEV v1.3) accredited wants to maintain the accreditation, they need to migrate to CMMI v2.0. Hence, the main objective of this proposal is to facilitate the migration from CMMI-DEV v1.3 to CMMI v2.0.

To address CMMI textual reference models being complicated and provide organizations a better understanding of CMMI v2.0, we propose a reference model of CMMI v2.0 using the Enterprise Architecture (EA) modelling language, ArchiMate. We represent both concepts and relationships of CMMI using ArchiMate's graphical elements.

To conduct this work, we chose the Design Science Research Methodology (DSRM). Therefore, the structure of this document is highly influenced by it. Section 2 explains the chosen research methodology. Section 3 contains the motivation for this work and the statement of our research problem. Section 4 is the theoretical background, where we describe the main concepts necessary to understand this research. Section 5 contains the related work, consisting of an analysis of already existing solutions related to this research's context. Section 6 is the design and development phase, where we present our proposal, well as the main objectives we want to achieve with its use. Section 7 is the demonstration of the proposed solution in a real-world organization. Section 8 is the evaluation of the proposed solution. Lastly, section 9 concludes with an overview of the work that was done.

2. Research Methodology

The approach chosen to guide this work was the DSRM. DSRM is an iterative methodology which combines principles, practices, and procedures required to carry a Design Science (DS) research. It provides guidance for research in Information Systems (IS) and other disciplines, as well as a mental model to present and evaluate DS research in IS [10, 19].

The main goal of DS in IS research is to create and evaluate IT artifacts intended to support the solution for the identified problems [19]. In this research, the artifacts that we are going to create and evaluate are models and constructs, by models we are referring to the metamodel of CMMI and by constructs, the mapping of CMMI v2.0 in ArchiMate [10, 19].

The DSRM process includes six phases [19] and this research conforms as follows:

1. **Problem identification and Motivation:** Difficulty to migrate from CMMI-DEV v1.3 to CMMI v2.0 considering the complicated CMMI textual reference models;
2. **Define the objectives of a solution:** Represent concepts and relations of CMMI v2.0 in ArchiMate, represent concepts and relations of CMMI v1.3 and v2.0, and allow users to navigate to any part of CMMI v2.0;
3. **Design and development:** Mapping of CMMI v2.0 in ArchiMate and respective Metamodel, CMMI v2.0 Reference model, and CMMI-DEV v1.3 to CMMI v2.0 visual practice mapping;
4. **Demonstration:** Mapping between company's procedures and CMMI-DEV v1.3, mapping between company's procedures and CMMI v2.0, and modelling the transition;
5. **Evaluation:** Wand and Weber method, Moody and Shanks quality management framework, questionnaire, and interview;
6. **Communication:** Paper and dissertation.

3. Research Problem

In today's high-technology business environment, an integrated approach to business and IT is indispensable to face the challenges of the changing global business scene [18].

SPI programs have attracted much attention in research and practice due to quality and reliability concerns, outsourcing opportunities, and expanding complexity, that result from marketplace demands [18].

Under these circumstances, the SPI framework CMMI has been widely promoted and was already used by over 10000 organizations from more than 100 countries all over the world [7].

Some companies find it a necessity to be CMMI accredited to negotiate and win contracts, others want economic and other benefits as a result. The benefits of implementing this framework include decreased costs, improved delivery schedule, productivity, quality, customer satisfaction, and increased ROI [7, 9].

However, although CMMI has clear benefits, only a small portion of companies adopt it. Studies done in companies in Australia [24], Malaysia [14], and China [11] show that organizations consider the program to be too costly, time consuming, and claim the lack of specialized personnel responsible for quality.

Many people reported that adopting CMMI can be quite complicated and often difficult. However, they also acknowledged that the returns from investing in this framework outweigh the expense of implementing it [12, 13].

One reason for the implementation of CMMI to take a lot of time and resources is that CMMI models are complicated. The textual reference models contain a lot of information and organizations need to adopt and integrate multiple practices. There are around two hundred practices in the models with several relationships between them, which makes it difficult to analyze and can be overwhelming for companies.

CMMI models are a set of best practices that focus on what needs to be done to improve performance and not how to do it [6]. For that reason, the existing textual reference models of CMMI-DEV v1.3 [6] and CMMI v2.0 [7] can be ambiguous. The reference models contain very extensive text, various technical concepts and numerous relationships between different practices. Additionally, there are many different concepts in the two most recent versions of CMMI.

With the release of CMMI v2.0, the architecture of CMMI was specifically designed to be flexible, agile, and evolve with the business, technology trends and market demands [7] but it is no less complicated for companies that are already CMMI-DEV v1.3 accredited and need to migrate to CMMI v2.0. If a company is CMMI-DEV v1.3 accredited, they need to migrate to CMMI v2.0 until September 30th 2020 [5], otherwise they will lose the accreditation.

We experienced, first hand, that companies want to delay the migration, as far as it is allow by the CMMI Institute, because of its difficulty.

By using EA models with graphical elements, like ArchiMate, we can provide a less complicated reference model that is more appealing to users and easier to analyze, thus minimizing the impact of a mandatory migration. This model can be helpful for consultants and quality assurance teams, as well as being useful in training sessions.

Therefore, based on the complicated textual reference models, the problem this research aims to solve is **the difficulty to migrate from CMMI-DEV v1.3 to CMMI v2.0.**

4. Theoretical Background

4.1. Capability Maturity Model Integration

Originally created for the U.S. Department of Defense to assess the quality and capability of their software contractors, CMMI models have expanded beyond software engineering to help any level of an organization in any industry. It now includes the entire product life cycle and focus on organizational development [2, 7, 15].

By providing a set of global best practices, this framework enables businesses to improve performance of their key business processes through building, improving and measuring their capabilities [7].

This research focus on the most recent version of CMMI, which is CMMI v2.0, and on the migration from CMMI-DEV v1.3. CMMI-DEV v1.3 is one of the Constellations of version 1.3 [6]. Now, in CMMI v2.0, these constellations were integrated into one single model.

In the CMMI v2.0 Model there are 229 Practices divided into 25 Practice Areas. These Practice Areas are split into 10 Capability Areas that are then divided into 4 Categories [7].

An organization can chose to achieve a maturity level or a capability level. In this research, we are focusing on maturity levels. These levels represent a staged path for an organization's performance and process improvement efforts based on predefined Practices. There are five maturity levels, each building on the previous ones by adding new functionality or rigor. The maturity levels cannot be skipped and a particular level is only achieved when all Practices belonging to that level (and all Practices belonging to lower maturity levels) have been successfully implemented [3, 7].

Between the two most recent versions of CMMI, there are significant changes in terms of structure and terminology [4]. For instance, Process Area is now called Practice Area, Process Area Category is now named Capability Area, Category and Practice Group (Level) are new terms that did not exist in the previous version, Generic Practices and Generic Goals have been replaced by two institutionalization Practice Areas, and, without Generic Practices, the Specific Practices designation is no longer needed.

4.2. Enterprise Architecture

The term EA refers to the architecture at the level of an entire organization, providing a holistic view of the enterprise. EA is defined by Lankhorst [15] 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" .

This type of architecture, captures crucial parts of both business and IT, helping to keep the business essentials, while allowing for maximum flexibility. A better alignment between business and IT leads to lower cost, higher quality, better time-to-market, and greater customer satisfaction. Hence, to have a successful business, a good architecture is needed [8, 15, 25].

EA has become an indispensable instrument in controlling the complexity of an enterprise structure, processes, and systems. Through architecture models, views, presentations, and analyzes, the communication gap between architects and stakeholders has been reduced [15, 25].

4.3. ArchiMate

ArchiMate, an Open Group Standard, is a modelling language for EA that provides a uniform graphical representation for diagrams that describe EAs [27].

The goal of using ArchiMate is to develop an architecture (and create views of the architecture) that describes, analyzes, and communicates stakeholder's concerns as they change over time [15, 27].

This language represents different architecture domains and their fundamental relations and dependencies. It allows to integrate concepts from different domains of business architectures due to a service-oriented approach that differentiates and relates different layers of an EA. The layers expose their functionality as a form of services to the layers above [15, 27].

5. Related Work

5.1. CMMI-DEV v1.3 Reference Model in ArchiMate

Valverde et al. [28] proposed a graphical reference model for CMMI-DEV v1.3 using ArchiMate as the chosen EA modelling language, to reduce the high perceived complexity of CMMI by its users.

This thesis focused on providing a visual representation of CMMI-DEV v1.3. Based on interviews, they believe they were able to lower the user's perceived complexity of CMMI, therefore contributing to turn the CMMI framework easier to use, allowing users to read and understand the CMMI framework more easily and in an interactive way.

This research is a very important contribution for ours, since we are doing a follow up from it. We will use the knowledge of this CMMI-DEV v1.3 reference model for the parts related with CMMI-DEV v1.3 in our work.

5.2. CMMI Ontologies

Other than in Valverde et al. thesis [28], there are no researches about CMMI being modeled in ArchiMate but there are other CMMI ontologies.

Soydan and Kokar [23] proposed a partial formalization of CMMI-DEV. It captures the definitions of a number of concepts of CMMI-DEV and relations among the concepts. The main purpose of this work was demonstrate an automatic determination of a maturity level based upon data of the software engineering processes used by an organization. Towards this aim, a comprehensive formalization of the CMMI-DEV model was expressed in the formal language OWL.

Musat et al. [17] proposed a Model Driven based tool to automatically generate a language that supports CMMI Process Areas specification. This tool provides a framework that lets the user translate the CMMI generic model into a domain specific model, automatically generating a Domain Specific Language with multiple possibilities of transformation.

5.3. Information Systems Frameworks in ArchiMate

The ArchiMate modelling language was already used to model different IS frameworks. We are going to highlight other researches done using this modelling language.

Teixeira et al. [26] proposed a metamodel of ISO 31000 in ArchiMate. The main objective was to reduce the perceived complexity of ISO 31000, thus facilitating the understanding of the standard.

Vicente et al. [29] proposed an ITIL business motivation model in ArchiMate. The goal was to enhance ITIL with a formal representation of its business motivation model. The result was a set of consistent models with the whole ITIL motivation.

Silva et al. [22] used ArchiMate to model TIPA, a process assessment framework. The goal was to enhance this framework through a EA related notation. The result was a graphical notation of the TIPA framework using the ArchiMate modelling language, which created a bridge between EA and TIPA.

Almeida et al. [1] used ArchiMate to assess COBIT 5 and ITIL implementations. The main goal of this research was to reduce the complexity of mechanisms for IT enterprise governance, by facilitating the assessment of these mechanisms when used simultaneously. The authors proposed a model in ArchiMate that demonstrates the similarity between the process assessment models for COBIT 5 and ITIL.

Finally, Percheiro et al. [20] proposed a way to represent the ITIL metamodel in ArchiMate, as well as its integration with the COBIT metamodel. The goal was to demonstrate that metamodelling is a useful technique to gain a theoretical foundation and integrate them.

We can take valuable information from these researches to formulate our proposal. These frameworks have concepts with similar meanings to CMMI concepts, thus we can use the ArchiMate elements the authors used to represent our concepts. Furthermore, these representations help to validate that representing these frameworks with an EA modelling language, like ArchiMate, can reduce complexity and improve communication.

6. Proposal

6.1. Objectives

The main objective is to facilitate the migration for companies that are already CMMI-DEV v1.3 accredited and need to migrate to CMMI v2.0 by providing a reference model in ArchiMate to help with this transition.

To achieve the main objective and, consequently, the solution of our problem, the following objectives need to be accomplished:

- The solution must represent all the main concepts and relationships of CMMI v2.0 and CMMI-DEV v1.3;
- The solution must represent all the main concepts and relationships of CMMI v2.0 in ArchiMate;
- The solution must allow users to navigate to any part of the CMMI v2.0 model.

6.2. Proposal Description

To address the problem identified in section 3, we propose a **CMMI v2.0 reference model**, designed in ArchiMate using the modelling tool BiZZdesign Enterprise Studio.

This research is a continuation of the research previously done by Valverde et al. [28], regarding a CMMI-DEV v1.3 reference model in ArchiMate. Therefore, we represent CMMI v2.0 also in ArchiMate, so that we have a common thread and can map the differences between the two versions of CMMI.

The chosen language was ArchiMate firstly because of the similarity between CMMI and ArchiMate concepts. ArchiMate allows us to describe and visualise our structure in a clear and simple way, thus anyone related to the business world will be able to understand an ArchiMate model. This language helps us draw a bigger picture, focusing on relationships instead of implementation details. ArchiMate is wider in scope than notations like UML or BPMN, which are domain-specific notations, but is less detailed [15, 27].

This proposal includes the following artifacts:

- Mapping of CMMI v2.0 in ArchiMate and respective metamodel;
- CMMI v2.0 reference model;
- CMMI-DEV v1.3 to CMMI v2.0 visual practice mapping.

6.2.1 Mapping of CMMI v2.0 in ArchiMate and respective Metamodel

To develop a metamodel for CMMI v2.0 using ArchiMate 3.0, we first mapped CMMI v2.0 concepts [7] to ArchiMate concepts [27], as shown in Table 1.

In addition to mapping the concepts, it is also important to describe the ArchiMate relationships used in the metamodel.

For the relationships between Categories, Capability Areas, Practice Areas, Practice Groups, and Practices, we chose ArchiMate's Composition relationship. This relationship indicates that an element consists of one or more concepts [27].

For the relationship between the Example Activities and the Practices, we chose ArchiMate's Realization relationship, since it represents what needs to be done in the organization in order to realize the Practice [7].

The Related Practice Areas are represented with ArchiMate's Serving relationship. Related Practice Areas represent the Practice Areas that provide something to other Practice Areas [7] and can be represented with the Serving relationship, since it models that an element provides functionality to another element [27].

As for the remaining relationships, they are represented with ArchiMate's Association relationship because those relations cannot be represented by any other relationship.

Based on the concepts and relationships that we chose to represent CMMI v2.0 in ArchiMate, we propose the **CMMI v2.0 metamodel** shown in Fig. 1.

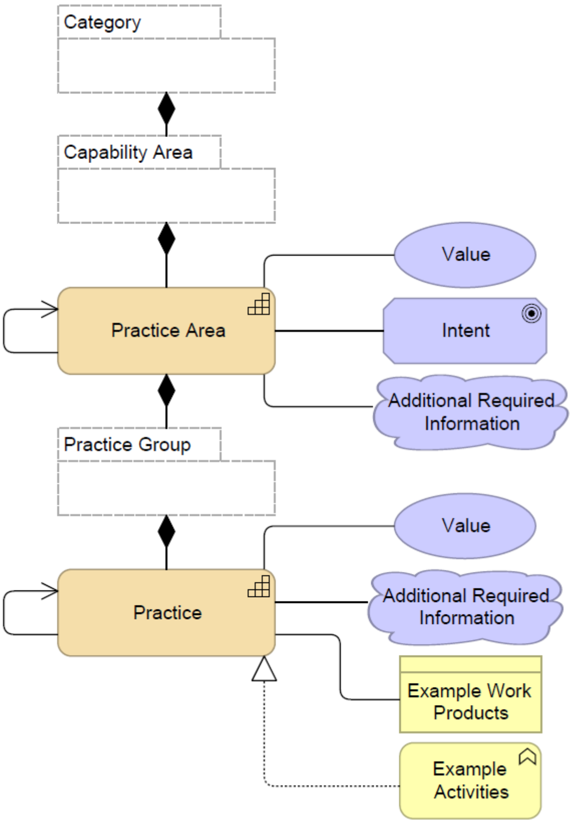

Figure 1: CMMI v2.0 Metamodel

6.2.2 CMMI v2.0 Reference Model

The **CMMI v2.0 reference model** is an instantiation of the metamodel presented in the previous section, with the information available in the CMMI Model v2.0 Manual [7].

In total, our proposed CMMI v2.0 reference model has 255 views. We were granted permission from the CMMI Institute to share part of this information. Fig. 2 shows part of the model's first level of abstraction.

As we can see in Fig. 2, the first level of abstraction contains the Category (Doing) and this Category contains four Capability Areas (Ensuring Quality, Delivering and Managing Services, Engineering and Developing Products, and Selecting and Managing Suppliers), each Capability Area contains a set of Practice Areas.

Inside each Practice Area, we have the concept

Table 1: Mapping of CMMI v2.0 in ArchiMate

CMMI	ArchiMate	Justification
Category Capability Area Practice Group	Grouping	These concepts can be defined as organizing structures, logical groups or types of views. Grouping element consists of a composition of concepts that belong together based on some common characteristic
Practice Area Practice	Capability	These CMMI concepts pretend to achieve a defined intent and value to the business. ArchiMate's Capability element represents an ability that an element possesses and defines what the business does or what it can do, it provides a high-level view of the current and desired abilities of an organization
Value	Value	CMMI's Value represents the business value we achieve by using that component. ArchiMate's Value represents the relative worth, utility, or importance of an element
Intent	Goal	Intent is the explanation of what results and accomplishments are expected as an outcome, it describes what the organization will achieve by satisfying a Practice Area. ArchiMate's Goal element represents a high-level statement of intent or desired end state for an organization
Additional Required Information	Meaning	Additional Required Information is important for clear understanding and interpretation of a Practice Area or Practice meaning. ArchiMate's Meaning element represents the interpretation of an element of the architecture
Example Work Products	Business object	Example Work Products are possible outputs of implementing processes that meet the intent of the Practice. ArchiMate's Business object can be used to represent information produced and consumed by a business process
Example Activities	Business function	Example Activities are possible actions that may be taken when implementing processes that meet a Practice's intent. ArchiMate's Business function element is a collection of business behavior based on a set of criteria aligned with an organization

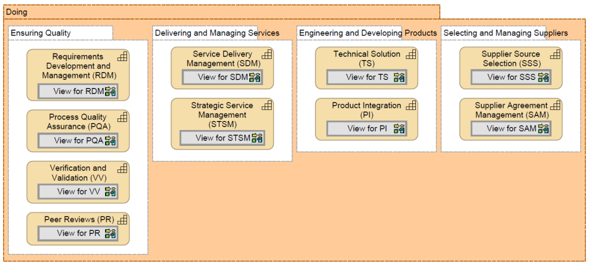
Doing
Ensuring Quality
Requirements Development and Management (RDM)
View for RDM
Process Quality Assurance (PQA)
View for PQA
Verification and Validation (VV)
View for VV
Peer Reviews (PR)
View for PR
Delivering and Managing Services
Service Delivery Management (SDM)
View for SDM
Strategic Service Management (STSM)
View for STSM
Engineering and Developing Products
Technical Solution (TS)
View for TS
Product Integration (PI)
View for PI
Selecting and Managing Suppliers
Supplier Source Selection (SSS)
View for SSS
Supplier Agreement Management (SAM)
View for SAM

Figure 2: Part of CMMI v2.0 Reference model ("Doing" Category)

of view. In this case, views help us navigate to other parts of the reference model and see an element in more detail. Therefore, if we double click the view, it will show us a Practice Area individually.

For instance, if we click on "View for VV", we can see the Practice Area Verification and Validation (VV) in more detail, as presented in Fig. 3.

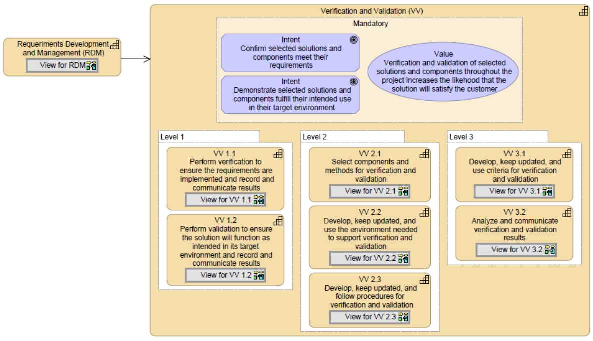
Requeriments Development and Management (RDM)
View for RDM
Verification and Validation (VV)
Mandatory
Intent
Confirm selected solutions and components meet their requirements
Intent
Demonstrate selected solutions and components fulfill their intended use in their target environment
Value
Verification and validation of selected solutions and components throughout the project increases the likehood that the solution will satisfy the customer
Level 1
VV 1.1
Perform verification to ensure the requirements are implemented and record and communicate results
View for VV 1.1
VV 1.2
Perform validation to ensure the solution will function as intended in its target environment and record and communicate results
View for VV 1.2
Level 2
VV 2.1
Select components and methods for verification and validation
View for VV 2.1
VV 2.2
Develop, keep updated, and use the environment needed to support verification and validation
View for VV 2.2
VV 2.3
Develop, keep updated, and follow procedures for verification and validation
View for VV 2.3
Level 3
VV 3.1
Develop, keep updated, and use criteria for verification and validation
View for VV 3.1
VV 3.2
Analyze and communicate verification and validation results
View for VV 3.2

Figure 3: Practice Area "Verification and Validation (VV)"

In this second level of abstraction of the reference model, we can see: the Practices that compose the Practice Area grouped in levels, which, in this case, go until Level 3; the Value of the Practice Area; the Intent of the Practice Area; and the Practice Areas that contribute to it, called Related Practice Areas. The Additional Required Information element, present in the metamodel (Fig. 1), is not represented yet, since in the CMMI Model v2.0 Manual [7] is still blank and will be added in the future.

Inside each Practice, we have again the concept of view, by clicking it, we are able to analyze each Practice individually. Due to this information not being public, CMMI Institute does not allow us to share it.

6.2.3 CMMI-DEV v1.3 to CMMI v2.0 visual practice mapping

The main objective of this proposal is to facilitate the migration for companies that are already CMMI-DEV v1.3 accredited and want to migrate to CMMI v2.0. Therefore, a visual mapping between CMMI-DEV v1.3 and CMMI v2.0 practices is useful.

CMMI Institute provided an Excel file that contained CMMI-DEV v1.3 to CMMI v2.0 practice mapping. Even though according to the Excel file CMMI-DEV v1.3 practices have direct correspondence to CMMI v2.0 practices, for some of the practices that correspondence is not complete. Thereby, following this Excel, we identified the changes that occurred in terms of Practice Areas and identified the gaps between CMMI-DEV v1.3 and CMMI v2.0 practices.

For instance, the Verification (VER) Process Area of CMMI-DEV v1.3 does not exist alone in CMMI v2.0. These Practices now equate to Practices in the Practice Areas VV and Peer Reviews (PR) of CMMI v2.0. Also, the Validation (VAL) Process Area of CMMI-DEV v1.3 does not exist alone in CMMI v2.0. These Practices now equate to Practices in the VV Practice Area of CMMI v2.0. Thereby, these two Process Areas of CMMI-DEV v1.3 merged into one new Practice Area of CMMI v2.0, VV, and the Practice Area PR was created.

Furthermore, even though most of the Practices have a direct correspondence between the two versions of CMMI, from VER and VAL to VV Practices, in CMMI v2.0 it was added to communicate the results of performing validation and verification activities, as well as communicate the results of analyzing those activities. Thus, the correspondence between some of the equivalent Practices is not complete. Also, from VER to PR, it was introduced in CMMI v2.0 to keep the procedures and materials updated, resolve the issues encountered during peer analysis and analyze the peer analysis results.

On the top of Fig. 4, we can see the VAL Specific Practices of CMMI-DEV v1.3 and, under it, the VV Practice Area of CMMI v2.0. The equivalent Practices are represented using circles of the same color. If a correspondence is not total, it is represented with a dashed circle.

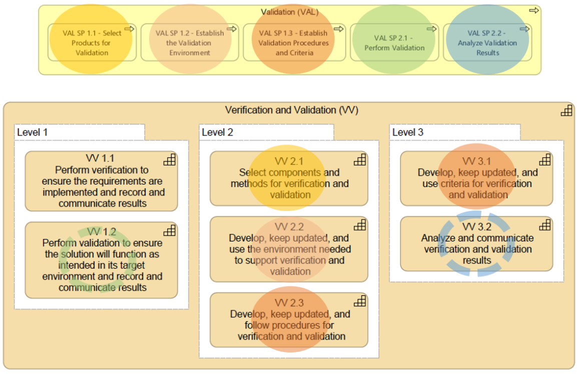
Validation (VAL)
VAL SP 1.1 - Select Products for Validation
VAL SP 1.2 - Establish the Validation Environment
VAL SP 1.3 - Establish Validation Procedures and Criteria
VAL SP 2.1 - Perform Validation
VAL SP 2.2 - Analyze Validation Results
Verification and Validation (VV)
Level 1
VV 1.1 Perform verification to ensure the requirements are implemented and record and communicate results
VV 1.2 Perform validation to ensure the solution will function as intended in its target environment and record and communicate results
Level 2
VV 2.1 Select components and methods for verification and validation
VV 2.2 Develop, keep updated, and use the environment needed to support verification and validation
VV 2.3 Develop, keep updated, and follow procedures for verification and validation
Level 3
VV 3.1 Develop, keep updated, and use criteria for verification and validation
VV 3.2 Analyze and communicate verification and validation results

Figure 4: Mapping between VAL (CMMI-DEV v1.3) and VV (CMMI v2.0)

7. Demonstration

To demonstrate that the solution we developed solves the identified problem and achieves the defined objectives, we applied it on a real world organization. This way, we were able to validate our proposal with the relevant stakeholders.

We chose a Portuguese IT company specialized in the development of banking software. This company is CMMI-DEV v1.3 Maturity Level 3 accredited and, until September 30th 2020 [5], needs to migrate to CMMI v2.0, to continue being CMMI accredited.

Our demonstration was done using the ArchiMate modelling language and the BiZZdesign modelling tool and comprises the following steps:

1. Map company's procedures to CMMI-DEV v1.3;
2. Map company's procedures to CMMI v2.0;
3. Model the transition.

Firstly, the company chose two process areas from CMMI-DEV v1.3 that, for them, were the most relevant. The chosen process areas were **VER** and **VAL**.

7.1. Mapping between company's procedures and CMMI-DEV v1.3

As proposed by Valverde et al. [28], the first step is to model the AS-IS of the organization's EA, using ArchiMate. Therefrom, we modeled, in ArchiMate, the four company's procedures that answered Practices from the VER and VAL Process Areas of CMMI-DEV v1.3.

The second step was to map the company's procedures to CMMI-DEV v1.3. We identified which actions satisfied the VER and VAL Specific Practices and represented it using circles of the same color. On the top of Fig. 5, we can see the VAL Specific Practices of CMMI-DEV v1.3 and, under it, the company's procedure that implements those Practices.

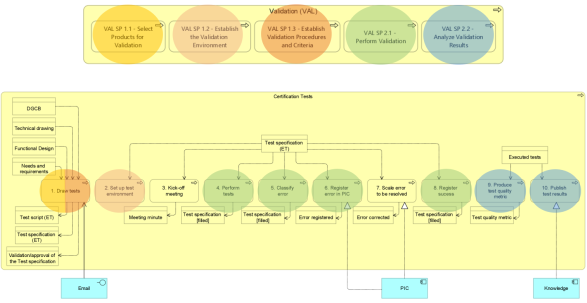
Validation (VAL)
VAL SP 1.1 - Select Products for Validation
VAL SP 1.2 - Establish the Validation Environment
VAL SP 1.3 - Establish Validation Procedures and Criteria
VAL SP 2.1 - Perform Validation
VAL SP 2.2 - Analyze Validation Results
Certification Tests
DGCB
Technical drawing
Functional Design
Needs and requirements
Test specification (ET)
Executed tests
1. Draw tests
2. Set up test environment
3. Kick-off meeting
4. Perform tests
5. Classify error
6. Register error in PIC
7. Scale error to be resolved
8. Register sucess
9. Produce test quality metric
10. Publish test results
Test script (ET)
Test specification (ET)
Validation/approval of the Test specification
Meeting minute
Test specification [filled]
Test specification [filled]
Error registered
Error corrected
Test specification [filled]
Test quality metric
Email
PIC
Knowledge

Figure 5: Mapping between VAL Specific Practices and company's procedure

8. Mapping between company's procedures and CMMI v2.0

The second phase consisted of mapping the company's procedures to CMMI v2.0 and identifying what Practices of CMMI v2.0 are not being answered in the company's current state (AS-IS).

On the top of Fig. 6, we can see the PR Practice Area of CMMI v2.0 and the respective Practices. Under it, we have the company's Peer Reviews procedure, that is the procedure in which these Practices should be answered.

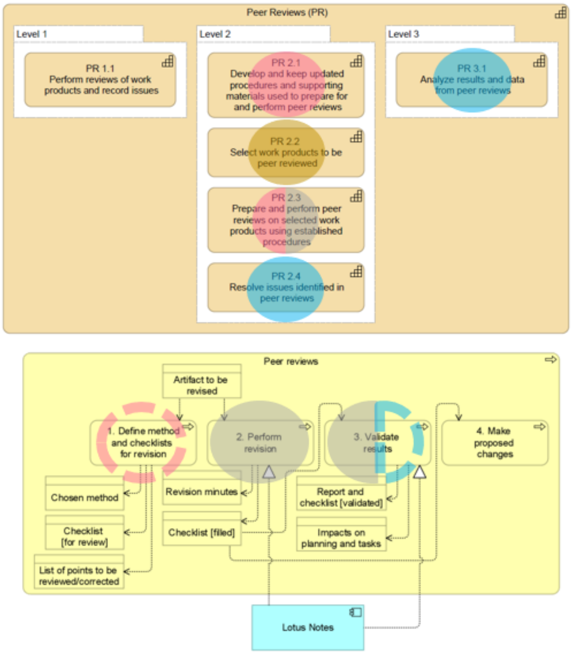
Peer Reviews (PR)
Level 1
Level 2
Level 3
PR 1.1
Perform reviews of work products and record issues
PR 2.1
Develop and keep updated procedures and supporting materials used to prepare for and perform peer reviews
PR 2.2
Select work products to be peer reviewed
PR 2.3
Prepare and perform peer reviews on selected work products using established procedures
PR 2.4
Resolve issues identified in peer reviews
PR 3.1
Analyze results and data from peer reviews
Peer reviews
Artifact to be revised
1. Define method and checklists for revision
2. Perform revision
3. Validate results
4. Make proposed changes
Chosen method
Revision minutes
Report and checklist [validated]
Checklist [for review]
Checklist [filled]
Impacts on planning and tasks
List of points to be reviewed/corrected
Lotus Notes

Figure 6: Mapping between PR Practices and Peer Reviews procedure

As we can see in Fig. 6, from the dashed circles and missing color, the company's current EA does not fully satisfy the Practices "PR 2.1" and "PR 2.3" and does not answer the Practice "PR 2.2". Therefore, the company needs to make changes in their procedures to fully satisfy these CMMI v2.0 Practices.

9. Modelling the transition

The third and final phase of this demonstration was to model the TO-BE of the company's procedures compliant with CMMI v2.0.

In the previous section, we identified the gaps between the two versions of CMMI. Therefrom, we modeled the desired state of the company's EA fully satisfying the VV and PR Practice Areas of CMMI v2.0, using ArchiMate's Migration viewpoint.

ArchiMate's Migration viewpoint is used to model the transition from an existing architecture to a target architecture [27].

In Fig. 7, we have this representation for the company's Peer Reviews procedure, where we can see the AS-IS procedures, as well as the TO-BE procedures. Our baseline was the company's EA compliant with CMMI-DEV v1.3 and our target was the company's EA compliant with CMMI v2.0. The transition was to identify the gaps, which we did in the previous section. The gap between our baseline architecture and our transition architecture was obtained by mapping the company's procedures with CMMI v2.0. The gap between our transition architecture and our target architecture was the processes and outputs needed for the procedure to be compliant with CMMI v2.0.

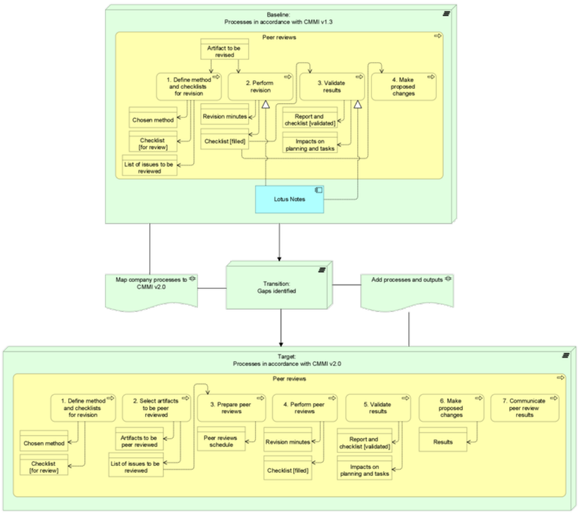
Baseline:
Processes in accordance with CMMI v1.3
Peer reviews
Artifact to be revised
1. Define method and checklists for revision
2. Perform revision
3. Validate results
4. Make proposed changes
Chosen method
Revision minutes
Report and checklist [validated]
Checklist [for review]
Checklist [filled]
Impacts on planning and tasks
List of issues to be reviewed
Lotus Notes
Map company processes to CMMI v2.0
Transition:
Gaps identified
Add processes and outputs
Target:
Processes in accordance with CMMI v2.0
Peer reviews
1. Define method and checklists for revision
2. Select artifacts to be peer reviewed
3. Prepare peer reviews
4. Perform peer reviews
5. Validate results
6. Make proposed changes
7. Communicate peer review results
Chosen method
Artifacts to be peer reviewed
Peer reviews schedule
Revision minutes
Report and checklist [validated]
Results
Checklist [for review]
List of issues to be reviewed
Checklist [filled]
Impacts on planning and tasks

Figure 7: Migration viewpoint of the Peer Reviews procedure

10. Evaluation

10.1. Wand and Weber method

Wand and Weber method allows us to compare two grammars and examine their ontological completeness and ontological clarity. We used this method to analyze the concept mapping between CMMI v2.0 and ArchiMate (section 6.2.1) by identifying ontological deficiencies [30].

We found that our mapping was complete and not redundant, and discovered instances of two deficiencies, overload and excess. However, they do not represent a major problem while modelling. The excess does not present any issue and the overload can be fixed by adding a property to the ArchiMate elements which allows to distinguish between the different CMMI concepts.

10.2. Moody and Shanks quality model framework

The Moody and Shanks quality model framework allows us to evaluate and improve the quality of data models. We used this framework to assess the quality of our model artifact regarding some quality factors [16].

Our metamodel was constructed respecting CMMI and ArchiMate specifications and includes the relevant concepts and relationships described in CMMI v2.0. With the questionnaire and the interview, we concluded that our model was complete, simple, flexible, understandable, correct, and implementable.

10.3. Questionnaire

A form of evaluating IS artifacts is to perform a quantitative analysis of our proposal, which results in a measured or perceived numeric value [21]. To achieve this, we used a questionnaire.

The questionnaire's goal was to validate the correctness and utility of our reference model. It was shared with CMMI professionals and practitioners and 19 responses were collected.

The questionnaire results were positive. The subjects believed that this is a good way to represent this framework and that graphical models, such as ours, are useful and facilitate the use of CMMI. This leads us to believe that our proposal is valid and tackles the complexity of CMMI.

10.4. Interview

Demonstrations are considered an early evaluation activity [21]. Following the demonstration done in a real-world organization, we interviewed the Quality Assurance Director and the Test Team Leader from that organization, to validate the value and correctness of our proposal, as well as verify if the demonstration of our proposal helped solve the identified problem.

The feedback given in the interview was very positive. For them, our CMMI v2.0 reference model is a good way to represent the framework and is especially useful in training sessions. The mappings with the colors and the dashed circles allowed them to clearly see what changes need to be done. Also, being able to see the whole model and the whole path, from beginning to end, makes it easier to follow and to understand the migration. Thereby, we can conclude that our proposal was useful and will help this company migrate from CMMI-DEV v1.3 to CMMI v2.0.

11. Conclusion

This research was done by following DSRM which comprises 6 phases of development. First, the research problem was identified as being the difficulty to migrate from CMMI-DEV v1.3 to CMMI v2.0. Then, we defined the main objective of our proposal, which is to facilitate the migration for companies that are already CMMI-DEV v1.3 accredited and want to migrate to CMMI v2.0 by providing a reference model in ArchiMate to help this transition.

To solve the problem identified, we did a concept mapping between CMMI and ArchiMate and developed a metamodel. We then instantiated the metamodel, creating our CMMI v2.0 reference model. Additionally, we did a visual mapping between CMMI-DEV v1.3 and CMMI v2.0 Practices.

The utility of our model was demonstrated in a real-world organization that was already CMMI-DEV v1.3 accredited and needed to migrate to CMMI v2.0. Our demonstration was done using the ArchiMate modelling language and the BiZZdesign modelling tool and consisted on mapping company's procedures with CMMI-DEV v1.3, mapping company's procedures with CMMI v2.0, and modelling the transition.

We evaluated our proposal using a questionnaire to experts and practitioners and an interview with stakeholders of the organization in which we did our demonstration, as well as other well-known techniques to evaluate DS artifacts.

With the collected feedback, we proved our proposal is easy to understand, easy to use, and useful for the organization, facilitating the migration, as well as making CMMI more understandable.

We intend to communicate our proposal to the scientific community by submitting a research paper to the ECIS 2020 conference.

As for limitations, since CMMI v2.0 is so recent, it was difficult to find different sources of information because there are not many researches about it yet. Regarding the demonstration, the company was going through changes in terms of procedures and tools and delayed the migration to CMMI v2.0, causing a delay in our work. Another limitation we found regards this research being a continuation of Valverde et al. [28]. CMMI-DEV v1.3 was more complex, in CMMI v2.0 some of the problems were tackled by simplifying the structure of CMMI. Therefore, although the graphical model is still useful to aid the use and understanding of CMMI v2.0, it is not as useful as it was for CMMI-DEV v1.3.

Regarding future work, it would be interesting to demonstrate and evaluate our proposal in more organizations, as well as do the demonstration for other CMMI practice areas. It would be useful to automate the mapping between CMMI Practices and the company's EA. Lastly, to show the advantages of migrating to CMMI v2.0, it would be interesting to show the impact of CMMI in a project compliant with CMMI-DEV v1.3 Practices versus the impact if it was compliant with CMMI v2.0 Practices.

References

[1] R. Almeida, P. L. Pinto, and M. M. da Silva. Using ArchiMate to Assess COBIT 5 and ITIL Implementations. *25th International Conference on Information Systems Development, Poland*, 2016.

[2] CMMI Institute. About CMMI® Institute, 2018.

[3] CMMI Institute. Appraisals, 2018.

[4] CMMI Institute. How Is The Terminology Changing In CMMI V2.0?, 2018.

[5] CMMI Institute. When Will CMMI V1.3 Sunset And No Longer Be Supported?, 2018.

[6] CMMI Product Team. *CMMI for Development, Version 1.3*. Software Engineering Institute, November 2010.

[7] CMMI Product Team. *CMMI Model V2.0*. CMMI Institute, 2018.

[8] U. Franke, M. Ekstedt, R. Lagerström, J. Saat, and R. Winter. Trends in Enterprise Architecture Practice – A Survey. *International Workshop on Trends in Enterprise Architecture Research*, page 16–29, 2010.

[9] D. Goldenson and D. Gibson. Demonstrating the Impact and Benefits of CMMI: An Update and Preliminary Results. *Carnegie Mellon University*, October 2013.

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

[11] D. B. Huang and W. Zhang. CMMI in Medium & Small Enterprises: Problems and Solutions. *2nd IEEE International Conference on Information Management and Engineering*, 2010.

[12] Jeff Dalton. Adopting the CMMI - How hard is it?, 2012.

[13] John Powell. Survey of the SEI / Carnegie Mellon CMMI framework, 2012.

[14] N. Khurshid, P. Bannerman, and M. Staples. Overcoming the First Hurdle: Why Organizations Do Not Adopt CMMI. *Springer*, 2009.

[15] M. Lankhorst. *Enterprise architecture at work: Modelling, Communication, and Analysis*. Springer, 2017.

[16] D. L. Moody and G. Shanks. Improving the Quality of Data Models: Empirical Validation of a Quality Management Framework. *Information Systems*, 28(6):619–650, 2003.

[17] D. Musat, V. Castaño, J. A. Calvo-Manzano, and J. Garbajosa. MATURE: A Model Driven bAsed Tool to Automatically Generate a lanGUage That suppoRts CMMI Process Areas specification. *European Conference on Software Process Improvement*, pages 48–59, 2010.

[18] G. O'Regan. *Concise Guide to Software Engineering: From Fundamentals to Application Methods*. Springer, 2017.

[19] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77, 2007.

[20] I. Percheiro, R. Almeida, P. L. Pinto, and M. M. da Silva. Towards Conceptual Meta-Modeling of ITIL and COBIT 5. *In European, Mediterranean, and Middle Eastern Conference on Information Systems*, pages 478–491, 2017.

[21] N. Prat, I. Comyn-Wattiau, and J. Akoka. Artifact Evaluation in Information Systems Design Science Research - A Holistic View. *PACIS 2014 Proceedings*, 23:1–16, 2014.

[22] N. Silva, M. M. da Silva, B. Barafort, M. Vicente, and P. Sousa. Using ArchiMate to Model a Process Assessment Framework. *In Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 1189–1194, April 2015.

[23] G. H. Soydan and M. M. Kokar. A partial formalization of the CMMI-DEV - A capability maturity model for development. *Journal of Software Engineering and Applications*, 2012.

[24] M. Staples, M. Niazi, R. Jeffery, A. Abrahams, P. Byatt, and R. Murphy. An exploratory study of why organizations do not adopt CMMI. *Journal of Systems and Software*, 80(6):883–895, June 2007.

[25] T. Tamm, P. B. Seddon, G. Shanks, and P. Reynolds. How Does Enterprise Architecture Add Value to Organisations? *CAIS*, 28:141–168, 2011.

[26] J. Teixeira, M. M. da Silva, and E. Carreira. Modelling Risk Management using ArchiMate, 2017.

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

[28] L. Valverde, M. M. da Silva, and M. R. Gonçalves. CMMI-DEV v1.3 Reference Model in ArchiMate, 2018.

[29] M. Vicente, N. Gama, and M. M. da Silva. Modeling ITIL Business Motivation Model in ArchiMate. *Business Informatics (CBI), IEEE 15th Conference*, pages 270–275, 2013.

[30] Y. Wand and R. Weber. On the Ontological Expressiveness of Information Systems Analysis and Design Grammars. *Information Systems Journal*, 3(4):217–237, 1993.