

A Path for the Implementation of Best Practices for Software Requirements Management Process Using a Multimodel Environment

Gloria Piedad Gasca-Hurtado[1] and Mirna Muñoz[2(✉)]

[1] Ingeniería en Sistemas, Facultad de Ingenierías, Universidad de Medellín, Carrera 87 no. 30-65, Medellín, Colombia
gpgasca@udem.edu.co

[2] Centro de Investigación en Matemáticas - Unidad Zacatecas, Parque Quantum, Ciudad del Conocimiento Avenida Lassec, Andador Galileo Galilei, Manzana, 3 Lote 7, CP 98160 Zacatecas, Zac, Mexico
mirna.munoz@cimat.mx

Abstract. Continuous improvement is one of the topics of interest for organizations seeking positioning opportunities in the world market. However, software development organizations have high levels of difficulty to implement best practices that address continuous improvement. This paper presents a path to follow to facilitate the work of continuous improvement in a software development organization and that seeks to implement best practices in the software requirements management process. The path is drawn from an analysis of software process improvement models and standards related to software development best practices, under a multimodel environment. The path is structured with a set of techniques, tools, activities, and outputs associated with identified best practices, to facilitate the implementation of improvements in the software requirements management process. Besides, the established path is proposed as an alternative to facilitate the process improvement using a multimodel environment, this way allows establishing balance and instances of collaboration among best practices independent of the model or standard to be implemented.

Keywords: Requirements management · Standards · Models · Multimodel · Implementation path · Software development process

1 Introduction

Globally, the software industry is made up of Small and Medium (SMEs) sized companies in around 92% [1]. At the technology level and specifically for software development companies, efficient software engineering practices, which are tailored to the size, resources, and type of business, are needed. This need arises because of the high effort involved in the implementation of models and standards [2] such as CMMI-DEV v 1.3 [3], ISO 12207 [4], ISO 15504 [5] and ISO 9004:2009 [6].

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12254, pp. 812–828, 2020.
https://doi.org/10.1007/978-3-030-58817-5_58

Research focused on determining the importance and difficulty of implementing models and standards for process improvements, confirms that they are oriented to large organizations, which represents less than 10% of the global software industry [7, 8]. Besides, these models and standards do not meet the needs of SMEs [7, 8]. Some of the causes that make difficult the implementation of these models and standards are the costs, the effort, and the return on investment (ROI) produced [9, 10].

The above mentioned, evidence mainly the Latin American context, in which not all software development organizations have the conditions to implement such models and standards. However, they need to stay in continuous improvement to achieve an adequate maturity level to produce quality software.

For SMEs starting in Software Process Improvement (SPI) initiatives represent a great investment and they often find that their improvement goals are not achieved in the short term, and sometimes with significant resource losses [11].

In this context, the process improvement based on multi-model environments aims to provide a harmonized and unified approach through the implementation of different models and/or standards within a SPI initiative [12] to get benefits [13] such as: a) unify a single improvement plan, b) focus the improvement proposal in the organization needs, instead of focusing it on the model or standard that will be implemented [14], and c) reduce the effort to implement a complete model or standard [14].

Because most researchers and the software industry define requirements management as a key process, and it is recognized as a key phase for successful project management [15, 16]. The goal of this paper is to provide a path to follow to facilitate the implementation of best practices in the requirements management process. The established path is proposed as an alternative to facilitate the process improvement using a multi-model environment being the main contribution of this paper.

This work is structured as follows: Sect. 2 establishes a baseline framework on requirements management and multi-model environments. Section 3 describes the research methodology that was applied to establish the implementation path based on the multi-model environment. Section 4 describes the structure of the path of implementation of best practices for the requirements management, and finally, Sect. 5 presents the conclusions and the future work.

2 Theoretical Framework

Requirements engineering can be understood as a process throughout the lifecycle of software development, even after the deployment of the project. However, requirements engineering can also be understood as an activity to manage changes in requirements [17].

The Requirements Management aims to capture, store, disseminate, and manage information [18]. In general, requirement management includes all activities related to version control and change control, as well as, the requirements tracking. Besides, indicators are highlighting that the failure of projects is the requirement management [17, 19].

This is not surprising given that the requirement management process has a great impact on the effectiveness of all software development processes [20]. A study in the

UK found that of 268 documented developmental problems, 48% represented problems related to software requirements [21]. Another study identified problems in software development organizations associated with requirements engineering activities [22]. This allows us to conclude that the phases associated with requirements engineering, such as requirement management are success factors in software development, having a high impact on the success of software development projects.

For these reasons, it is important to make proposals and research that will help improve the way organizations implement requirements engineering processes [15].

Around the best practices, some researchers focus on defining them, as well as the frameworks, standards, and models that contain them such as: CMMI-DEV v1.3 [3], PRINCE2 [23], ISO/IEC 15504 [5], IEEE 830 [24], IEEE 1233, ISO 12207 [4], ISO 9000:2000 [25], PMBOK [26].

There are research initiatives aimed at facilitating the implementation of best practices in software development organizations related to software requirements. Within these initiatives exist a guide generated through a research study in Malaysia. This research was aimed to document the experience of IT Managers to standardize a guide for software projects in the Public Sector in Malaysia [27]. This proposal is complemented with requirements management tools that incorporate the best practices in requirement management [28].

GlobReq is a framework for the requirements engineering process of global software development projects. This proposal focuses on adapting and customizing a framework of requirement practices defined by Sommerville, focused on global software development projects [29].

A case study on requirement practices in three open-source software projects is another of the initiatives associated with the implementation of requirement practices to achieve successful software development projects. The objective of this study was focused on understanding how each project manages requirements and identify research lines on requirements engineering by proposing a taxonomy to describe the spectrum of formal requirements practices [30].

Besides, there are hybrid approaches between agile methodologies and traditional methodologies that are proposed taking into account the importance of requirements engineering as fundamental elements for software development [31]. In the same orientation of the agile methodologies, there are studies related to the evaluation of the requirements engineering process and the aspects that seem to be ignored when using an agile environment. From this study, we can outline a path related to the essential requirements engineering practices and their possible adaptation in the context of agile ideas such as simplicity, short notes, continuous validation, and frequent refactoring, among others [32].

There are other studies related to requirements engineering where requirements management measures are proposed as efficient predictors of stability and volatility of requirements and management of change requests. This proposal has been validated using ten of the proposed measures and the generation of an industrial case where it is pretended to validate the efficiency of the proposed predictions [33].

Other important initiatives have been defined in terms of the need to generate requirements efficiently, eliminating ambiguity aspects. For the above, the process of

generating requirements and its related activities are analyzed. It also focuses on methodologies, models, and requirements engineering techniques [34].

These studies not only demonstrate the interest of research and industry for the requirements engineering. It also shows that commercial software companies are looking for profits by actively exploiting viable business opportunities. To achieve and maintain a competitive advantage these companies face great challenges due to technological changes [35]. From the perspective of maximizing return on investment (ROI), Software development companies must focus on identifying and implementing the most cost-effective functionality, a goal that is strongly correlated with the identification of requirements that meet customer needs [36].

Unfortunately, this relationship seems to be underestimated: during the software engineering process, the requirements are not necessarily identified, qualified, or prioritized, which constitute weaknesses associated with the requirements management [15, 16, 37].

While these weaknesses can be solved implementing frameworks and standards such as CMMI-DEV v1.3, PRINCE2, ISO/IEC 15504, IEEE 830, IEEE 1233, ISO 12207, ISO 9000:2000, PMBOK, organizations increasingly tend to use a set of international standards and models to manage software development processes. They seek to increase customer satisfaction, achieve a competitive advantage, and benefit in process performance and normative compliance [38]. For this reason, a company probably prefers to implement a set with more than one software process improvement model, standard or normative [39]. This trend is known as a multimodel environment, an approach of the Software Engineering Institute (SEI) developed to harmonize the process improvement models [40]. However, there are difficulties in successfully implementing process improvement using multimodel environments, mostly due to the complexity of their implementation [41].

It is important to recognize that the implementation of a model or standard by itself or in a multi-model environment, according to the current trend of the organizations, it has an intrinsic complexity. For this reason, it is necessary to define a path to follow from the analysis of the needs of an organization and the process to be implemented.

3 Research Methodology

The design of activities is fundamental for applied disciplines. Design-oriented research has a long history in many fields, including architecture, engineering, education, psychology, and fine arts [42].

The area of Technology and Information Computing (TIC) has adopted many of the ideas, concepts, and methods of science based on the design that have originated in these other disciplines. This research paper has taken as a methodological basis the research paradigm proposed by Hevner [43]. This proposal consists of three cycles for scientific research based on design (See Fig. 1).

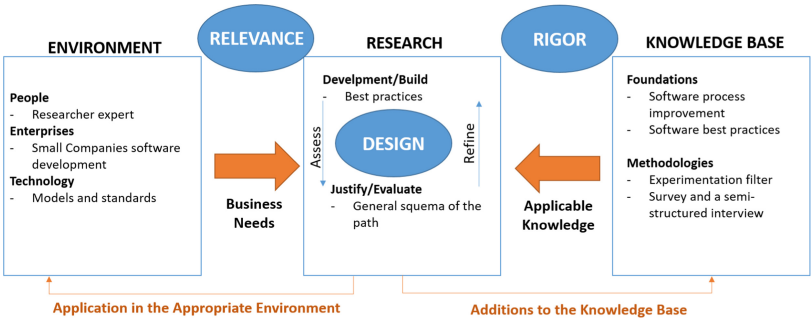

Fig. 1. General Scheme research paradigm proposed by Hevner [43].

The three cycles were followed to design the proposed path for implementing software development best practices for the RM process. Each one of these cycles was implemented as follows:

- In the *relevance cycle* the standards and models that will compose the multi-model environment, which will be used as a framework for the path, were identified. In this phase, the Models and Standards Similarity Study method (MSSS) [13] was used as a support strategy to establish a multimodel environment. As the method suggests, the models and standards were selected taking into account the application domain, and the problems and opportunities of the requirements management.
- In the *design cycle*, the base best practices, which are contained in the general schema of the implementation path, were selected. The selection of the best practices is based on the number of coincidences and relationships between the models and standards identified in the *relevance cycle*. A nomenclature was defined to identify relationships among different models and standards. Besides, the structure of the implementation path is established from the identified coincidences and relationships.
- In the *rigor cycle*, it is established the way to validate the implementation path. This validation was designed to be used by experts in process improvement. This cycle aims to establish a knowledge base that is suggested by the research paradigm.

4 A Path for Software Requirements Management

Next, the results obtained from the application research paradigm are shown.

4.1 Relevance Cycle

The MSSS method consists of a sequence of steps to determine how a reference model or standard is complemented by other models or standards.

Table 1 shows the steps of the MSSS method. Besides, it includes the adaptations done to the method and its justification.

Table 1. Adaptation of the MSSS method.

MSSS method steps	Adaptations	Justification
Select models and standards	Select the process to be compared	To focus the study on the process and select the best practices to be included in the implementation path
Select a reference model	Select models and standards	To focus on the selection of models and standards based on the focused process to define the implementation path
Select the process to be compared	Select a reference model or standard	A model or standard should be selected as a reference to integrate the multimodel environment based on specific criteria
Identify similarities between models		Without changes or adaptations from the original method
Create a matching template		
	Set the detail level	To facilitate the analysis of the information according to the recommendations of the MSSS method
Show obtained results		Without changes or adaptations from the original method

The results of the relevancy cycle allowed establishing the application domain, the problems, and the opportunities related to the requirement management. The following are the most relevant results found in this cycle:

1) The baseline for determining the similarity is the last level of reference model (CMMI-DEV), i.e. at the specific practice level. Making a comparative analysis at the specific practice level allows covering most of the structure of the reference model compared with other standards and models selected.
2) The correspondence template details the inputs, subpractices, and work products of the selected standards and models to complement the reference model [44].
3) Following the recommendations of the MSSS method, a glossary is established, which for this research is called nomenclature based on the initials of the model or standard, followed by a number as follows: CMMI-DEV v1.3 (C); IEEE 1233(I1); PRINCE2(P); ISO 12207 (IS1); ISO/IEC 15504 (IE1); ISO 9000:2000 (IS9); IEEE 830 (I8); PMBOK (PM)
4) The consolidated results describe how the evaluated elements of the models and standards complement the reference model. An example of the consolidated results is shown in Table 2 results table for the specific practice to establish the requirements for the product and the product components. The full analysis details the similarities, as well as the direct and indirect relationships established from the analytical study [44].

The similarity study performed allowed identifying the indirect relations of the elements of each standard and model analyzed, in Table 2 for this type of relations are used (α).

Table 2. Results table for the SP 2.1 Establish the requirements of the products and the product components.

SG 2 Develop the product requirements		
SP 2.1 Establish the requirements for product and product components		
Inputs	*Tools and techniques*	*Work products*
Customer Satisfaction (C) Business satisfaction (C) Project objectives and associated attributes (C) Production, operation and withdrawal requirements (C) Change requests (IE1) Customer petition (IE1) Customer requirements (IE1) Interaction that the software will have with people, hardware, other hardware, and other software (I8) (α) The speed, availability, response time, recovery time of various software functions, etc. (I8) (α) Portability, accuracy, maintenance, security requirements and other considerations (α) Standard requirements, application language, database integrity policies, resource limits, operating environment (I8) (α)	T2.1.1 Quality Function Deployment QFD (C) T2.1.2 Patterns of architecture (C) T2.1.3 Formal languages (IE1) T2.1.4 Traceability Matrices (IE1) T2.1.5 Prototypes (IE1) T2.3.1 Quality characteristics of the software described in the standard ISO/IEC 9126 (IE1) (α)	Derived requirements (C, IE1) Product requirements (C) Product component requirements (C) Architecture requirements, which specify or restrict the relationships between product components (C) Communication records (IE1) Change control records (IE1) System Requirements (IE1) Analysis report (IE1) Traceability records (IE1) Product functions (I8) (α) User characteristics (I8) (α) Restrictions that include regulatory policies, hardware limitations, interfaces to other applications, parallel operation, audit functions, control functions, language requirements, reliability requirements (I8)(α) Credibility of the application and security considerations (I8) (α) Specific requirements at a level of enough detail to facilitate design (I8) (α) Dependencies or statement of factors affecting the requirements stated in the SRS (I8) (α) Requirements not established by the customer but necessary for the specified use or intended use (IS9) (α)

Performing the similarity analysis, a complete coincidence is found between the models CMMI-DEV v1.3 and the ISO/IEC 15504 standard for inputs, tools, techniques, and work products. However, there are also indirect relationships (α) with the IEEE 830 and ISO 9000:2000 standards that can be identified by analyzing in detail

their input and work product elements. Because they allow defining requirements expressed in technical terms that can be used for design decisions.

4.2 Design Cycle

This cycle is composed of two activities: 1) selection of best practices, and 2) structure the path. Each activity was developed to obtain the results expected in the design phase. That means the schematic design of an implementation path for best practices as next described:

Activity 1. Selection of Best Practices
To define and select the best practices, the following criteria were considered:

$$P >= 3 \rightarrow BP \in R^1 \quad (1)$$

$$P < 3 \rightarrow BP \notin R^1 \quad (2)$$

Where, P = relation established in the consolidated results tables, BP = best practice and R1 = Primary implementation path.

For each one of the best practices that were eliminated by criteria *(2),* an analysis is performed. This analysis is carried out by a focus group of researchers who study each best practice regarding the model or standard to which the practice originally belongs. This study includes a comparison and identification of the elements that allows defining if the best practice is included in the path by criteria *(1)*, or it can be grouped as an activity or tool in R^1.

From this analysis, we obtain a first version of the implementation path, called the primary path (R^1), which the best practices base to implement and improve software requirements management, are identified. An example of a primary path for requirements management is shown in Fig. 2.

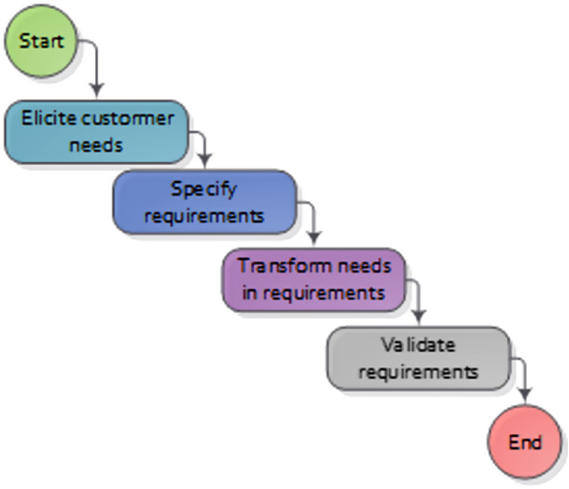

Fig. 2. General Scheme R^1 to implement best practices for requirement management.

The scheme shown in Fig. 2 corresponds to R^1. Up to this point, the scheme is sequential, indicating the start and end of the requirement management process. In the traced path, direct and indirect coincidences found in the similarity analysis are considered. These coincidences allow identifying both the standard and best practices according to the criteria for determining their membership in R^1. However, the objective of establishing a path for processes implementation, such as the requirement management, is to analyze and consolidate a multimodel environment, therefore the results were consolidated by applying the criteria *(1)* and *(2)*. Table 3 shows the consolidation done by taking the reference model as a base.

Table 3. Consolidated results by applying the criteria *(1)* and *(2)*.

CMMI-DEV v 1.3		P	IE1	I8	I1	PM	I9	IS12
Goal*	Practice							
DR	Eliciting needs	Y	Y	Y	Y	Y	Y	Y
	Transform needs into requirements	N	N	N	N	N	N	N
DPR	Establish products and components	N	Y	N	N	N	N	N
	Assign requirements to components	N	N	N	N	N	N	N
	Identify requirements interfaces	N	Y	N	N	N	N	N
AVR	Establish operating scenarios	N	N	N	N	N	N	N
	Establish a definition of quality and functionality	Y	N	N	N	Y	N	N
	Analyze requirements	N	N	Y	N	N	Y	Y
	Analyze requirements and balance	N	Y	N	N	N	N	Y
	Validate requirements	N	Y	Y	N	N	N	Y
RM	Understand	N	N	N	N	N	N	N
	Get commitment	N	N	N	N	N	N	N
	Manage changes	N	N	N	N	Y	Y	Y
	Maintain traceability	N	Y	Y	N	N	N	N
	Secure alignment	N	N	N	N	N	N	N

* DR: Develop requirement, DPR: Develop product requirement, AVR: Analyze and validate requirements and RM: Requirement Management.

As Table 3 shows, the reference model is CMMI-DEV v1.3 model, this model details the specific practices that must be considered to implement the requirements development (RD) and requirements management (REQM) processes. These processes have been grouped to select best practices in R^1. As shown in the table, it is recognized the existence of a direct relationship of all the standards and models to the practice of *eliciting needs*. This relation belongs to R^1 following the criteria *(1)*. This is the only practice that has total coincidence among the analyzed models and standards.

With the direct relation analyzed in the first instance, it is also identified that the CMMI-DEV v 1.3 model and the ISO/IEC 15504 standard focus on improving the IT processes, and they define the specific practices to implement requirements management. Besides, they present at a high level of detail the inputs, tools, techniques, and work products. These relationships were excluded from R^1, but documented to perform an analysis in the later phases for structuring the path.

From Table 4, it can also be determined that the PRINCE2 and PMBOK define the requirements management in the definition and analysis phase of the project scope, in the definition of the functionality and the quality attributes, as well as in change management. For this reason, they were excluded from R^1.

Activity 2. Structuring the Best Practices Implementation Path

From the primary path defined in activity 1, the techniques and tools that will complement the scheme of the primary path (R^1) are analyzed. A new analysis is performed considering relationships documented in the relevance cycle. Moreover, a new structure of the best practices implementation path is defined. This new structure complements the linear scheme (R^1) becoming a scheme by quadrants.

The selected scheme allows visualizing the linear overview of the primary path including four essential characteristics to implement best practices in each quadrant, including: a) best practices, b) key activities, c) tools and techniques, and d) process outputs.

To identify each quadrant, four symbols are defined as shown in Table 4.

Table 4. Identifier Symbols and its description

Identifier Symbol	Identifier description
	*Best practice*: a sequential alphanumeric nomenclature established with the prefix BPi. Where i is a numerical value.
	*Activity*: a sequential alphanumeric nomenclature established with the prefix A followed by a dot and the identifier corresponding to the best practice, e.g. A.BPi.-j. Where j is a numerical value.
	*Tool*: a unique alphabetical nomenclature established for the quadrant that identifies the set of tools for the corresponding BP. This nomenclature is followed by the suffix corresponding to BP, e.g. H.BPi.
	*Outputs*: a unique alphabetical nomenclature established for the quadrant that identifies the set of outputs resulting from the implementation of the corresponding BP. This nomenclature is followed by the suffix corresponding to BP, e.g. S.BPi.

Each scheme represents the best practices associated with the analysis of the recommendations of the multimodel environment used to define the best practices implementation path. In the scheme of the BP1 (Fig. 3) each quadrant is represented with their corresponding identifier. Quadrants characterize each best practice as follows:

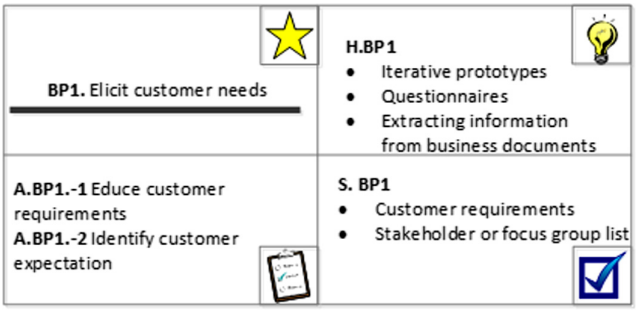

Fig. 3. Scheme for the practice (BP1) using the four essential characteristics.

Quadrant 1. Identified with the symbol of a yellow star. It corresponds to the name of the practice given from the multimodel environment.
Quadrant 2. Identified with the notepad symbol. It provides a set of activities that guides the implementation of the best practice provided in quadrant 1.
Quadrant 3. Identified with the bulb symbol. It provides a list of the tools that help in the implementation of the activities grouped in quadrant 2.
Quadrant 4. Identified with a tick square. It provides the outputs that represent the possible artifacts that are generated when executing the activities grouped in quadrant 2. Most of the time they are outputs resulting from the use of the techniques from quadrant 3.

Figure 4 shows the complete structure of the path by quadrants.

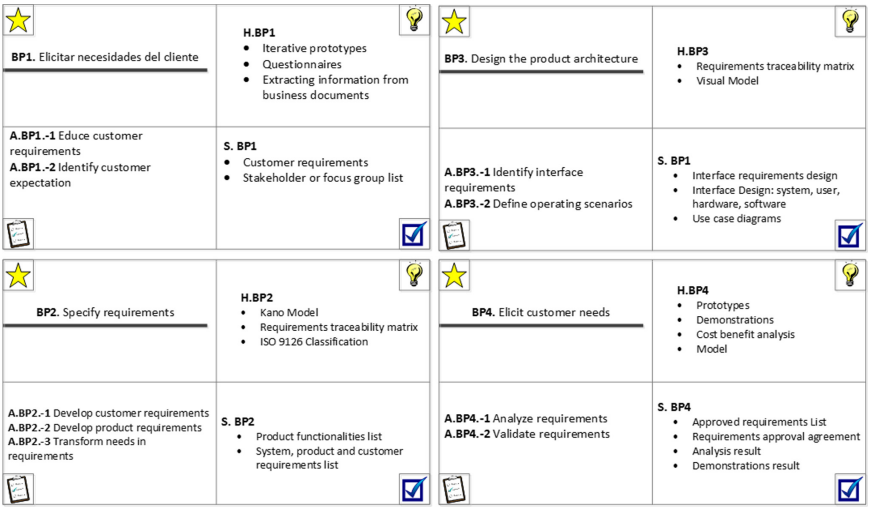

Fig. 4. Structure of the implementation path of requirement management by quadrants.

So far, these diagrams show activities, techniques, or methods and artifacts resulting from the implementation of best practices.

This proposal also includes a path in terms of flow diagram summarizing activities. This diagram is shown in Fig. 5. This path is defined with the identified activities in each one of the quadrants. Besides, the most important deliverables defined in the quadrants are included.

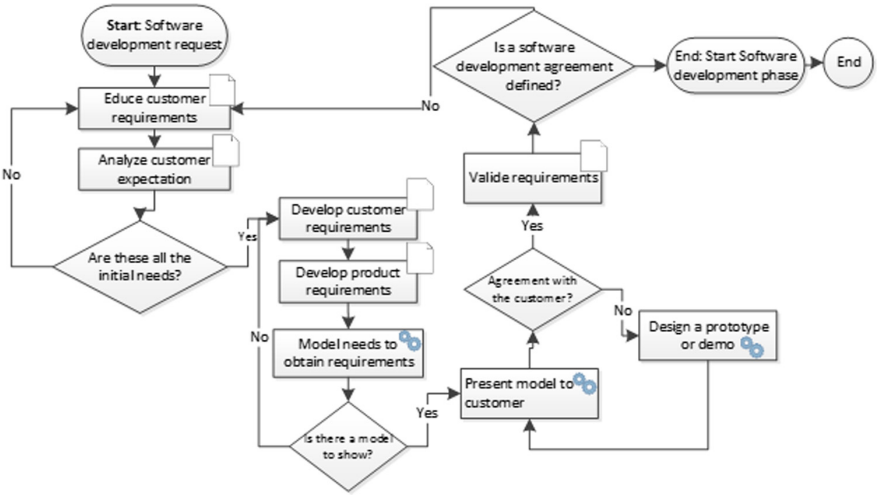

Fig. 5. Flow diagram for the implementation path.

4.3 Rigor Cycle

The validation process of the path was defined as an experimentation filter. This filter aims that researchers in the area present their opinion on the proposal. This phase was carried out using a survey and a semi-structured interview as an information collection technique.

Based on the similarities of the models and standards analyzed that address the requirements management process, the research team proposed the following recommendations:

- The requirements management process is the starting point in the implementation of process improvement for software development because it is the process that must be followed in detail to ensure an early quality.
- Software development organizations that implement best practices for software development focused on the specific practices of the RD and REQM process areas of the CMMI-DEV v1.3 model have complementary elements of other models, for example, techniques and tools. These complementary elements must be considered.
- The specific practices SP 1.1 and SP 3.2 of the RD process area, as well as the specific practice SP 1.3 of REQM, are critical in the process, therefore, they must be

implemented following in detail the recommendations of the CMMI-DEV v1.3 model. Besides, such criticality indicates that they are the first practices that must be implemented for the improvement of the software development process. Following some of the specific practices of greater criticality, contributes to reducing possible causes of failure associated with failures in definition, lack of participation from the user, incomplete requirements, and not properly managed changes [12].

- For the specific practices SP 2.1 and 2.3 of the RD process area of the CMMI-DEV v1.3 model, indirect relationships are identified $^{(\alpha)}$ relevant for the process improvement. These relationships reveal the importance of its implementation for each one of the analyzed models. Therefore, it is important that organizations seeking to implement improvements in processes related to software development requirements, consider such practices in the second instance.
- For specific practices SP 1.1, SP 2.2, SP 3.1, SP 3.3, SP 3.4, SP 3.5 of the RD process area and specific practices SP 1.1, SP 1.2, SP 1.4, SP 1.5 of the REQM process area of the model CMMI-DEV v1.3, it is concluded from the analyzes, that they are complementary practices because not all the analyzed models and standards are explicitly addressed them. However, the researchers' team considers it important to take into account that they are part of the process improvement and can be implemented in the medium-term.

4.4 Research Limitation

The experimentation filter conducted as a validation process can present threats to the results validation. Regarding to obtained results from the expert assessment can present threats associated with the experience and subjectivity of the experts in the study area, i.e. similarities of the models and standards analyzed that address the requirements management process.

To avoid the researcher bias during the validation process a supervised review process with a more experienced researcher was implemented. Moreover, we created a template format to register the semi-structured interview with the researchers, to obtain a baseline of the information from the interviewees.

Although, experts were chosen using the selection convenience technique, taking into account the experience of the experts in issues associated with the study area, such as: software engineering and software process improvement. For reducing the threats related to experience in the study area, we select the experts with high productivity in academy and research, using researchers databases.

In the future, we hope to collect results with a larger expert group, and compare the obtained results in a systematic literature review prepared for validation purposes. Such comparison will allow us to confirm the expert opinions and reduce the subjectivity of this validation.

5 Conclusions

An analysis of similarities between models and standards has been developed including CMMI-DEV v1.3, PRINCE2, ISO/IEC 15504, IEEE 830, IEEE 1233, PMBOK, ISO 9000:2000, and ISO 12207, all of them address the requirements management process.

The obtained results constitute a guide for the implementation of the best practices for the Requirements Management process. These best practices could be selected and implemented by small and medium-sized software development companies according to their needs, cost constraints, effort, and budget.

Besides, the result obtained allows determining the similarity or correspondence between the standards and models establishing the completeness, robustness, and detail for the implementation of standards and models widely accepted and adopted in small or medium organizations.

As a result of analyzing the coincidences and relations accomplished among the selected models and standards, it was possible to draw an implementation path. This path includes four basic practices for the development and management of project requirements. The implementation path was described in different charts such as, linear diagram for the basic path (see Fig. 2), four independent diagrams describing activities, methods or techniques, and outputs (see Fig. 4), and a flow diagram to explain the sequence of the path (see Fig. 5).

As future work, there has been identified that new lines of research associated with the design of best practices for other improvement processes. This would allow complementing the management path for software development requirements with intercommunicated processes that address the implementation of other software engineering processes.

The validation of this path is a line of work necessary to know results regarding the implementation of best practices grouped in the defined path and improvements that complement the design of the established implementation path.

References

1. Laporte, C.Y., Muñoz, M., Mejia, J., O'Connor, R.V.: Applying software engineering standards in very small entities-from startups to grownups. IEEE Softw. **35**(1), 99–103 (2017)
2. Kalinowski, M., et al.: Results of 10 years of software process improvement in Brazil based on the MPS-SW model. In: 2014 9th International Conference on the Quality of Information and Communications Technology (QUATIC), pp. 28–37 (2014)
3. Software Engineering Institute. CMMI for Development, Version 1.3 CMMI-DEV 1.3 (2010)
4. ISO/IEC. Systems and software engineering—Software life cycle processes, ISO/IEC 12207, ed. (2008)
5. ISO/IEC. International Standard ISO/IEC 15504 Software Process Improvement and Capability Determination, ed. (2004)
6. ISO/IEC. ISO 9004:2009, Managing for the sustained success of an organization – A quality management approach (2009)

7. Leung, H.K.N., Yuen, T.C.F.: A process framework for small projects. Softw. Process Improv. Pract. **6**(2), 67–83 (2001)
8. Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P., Murphy, R.: An exploratory study of why organizations do not adopt CMMI. J. Syst. Softw. **80**(6), 883–895 (2007)
9. Rico, D.F.: ROI of Software Process Improvement: For Project Portfolio Managers and PMO's (2004)
10. García, J., De Amescua, A., Velasco, M.: TOP 10 de factores que obstaculizan la mejora de los procesos de verificación y validación en organizaciones intensivas en software. Revista Española de Innovación, Calidad e Ingeniería del Software **2**, 18–28 (2006)
11. Zarour, M., Abran, A., Desharnais, J.-M., Alarifi, A.: An investigation into the best practices for the successful design and implementation of lightweight software process assessment methods: a systematic literature review. J. Syst. Softw. **101**, 180–192 (2015)
12. Srivastava, N., Singh, S., Dokken, T.: Assorted chocolates and cookies in a multi-model box. In: SEPG 2009 North America Conference. Software Engineering Institute, March 2019
13. Cuevas, G., Mejía, J., Muñoz, M., San Feliu, T.: Experiencia en la Mejora de Procesos de Gestión de Proyectos Utilizando un Entorno de Referencia Multimodelo. Revista Ibérica de Sistemas y Tecnologías de Información **1**, 87–100 (2010)
14. Muñoz, M., Mejía, J.: Establishing multi-model environments to improve organizational software processes. In: Rocha, Á., Correia, A., Wilson, T., Stroetmann, K. (eds.) Advances in Information Systems and Technologies, vol. 206, pp. 445–454. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36981-0_41
15. Mustapha, A.M., Arogundade, O.T., Misra, S., Damasevicius, R., Maskeliunas, R.: A systematic literature review on compliance requirements management of business processes. Int. J. Syst. Assur. Eng. Manag. **11**(3), 561–576 (2020). https://doi.org/10.1007/s13198-020-00985-w
16. Aguilar, J.A., Zaldívar-Colado, A., Tripp-Barba, C., Espinosa, R., Misra, S., Zurita, C.E.: A survey about the impact of requirements engineering practice in small-sized software factories in Sinaloa, Mexico. In: Gervasi, O., et al. (eds.) ICCSA 2018. LNCS, vol. 10963, pp. 331–340. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-95171-3_26
17. Xie, Y., Tang, T., Xu, T., Zhao, L.: Research on requirement management for complex systems. In: 2010 2nd International Conference on Computer Engineering and Technology, Chengdu, pp. V1-113–V1-116 (2010)
18. Paetsch, F., Eberlein, A., Maurer, F.: Requirements engineering and agile software development. In: WET ICE 2003. 2003 Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Linz, Austria, pp. 308–313 (2003)
19. Oza, N.V., Hall, T.: Difficulties in managing offshore software outsourcing relationships: an empirical analysis of 18 high maturity Indian software companies. J. Inf. Technol. Case Appl. Res. **7**, 25–41 (2005)
20. Sommerville, I., Ransom, J.: An empirical study of industrial requirements engineering process assessment and improvement. ACM Trans. Softw. Eng. Methodol. **14**, 85–117 (2005)
21. Beecham, S., Hall, T., Rainer, A.: Software process improvement problems in twelve software companies: an empirical analysis. Empir. Softw. Eng. **8**, 7–42 (2003). https://doi.org/10.1023/A:1021764731148
22. Damian, D.E., Zowghi, D.: RE challenges in multi-site software development organizations. Requirements Eng. **8**, 149–160 (2003). https://doi.org/10.1007/s00766-003-0173-1
23. OGC: Managing Successful Projects with PRINCE2 (2005)
24. IEEE: IEEE 830, Recommended Practice for Software Requirements Specifications, ed. (1998)

25. International Organization for Standardization. (Enero 16, 2013). ISO 9000: 2005 -Basic concepts and language (2005)
26. IEEE: A Guide to the Project Management Body of Knowledge, IEEE Guide (2004)
27. Haron, M., Harun, M., Mahrim, N., Sahibuddin, S., Zakaria, N.H., Abdul Rahman, N.: Understanding the requirement engineering for organization: the challenges. In: 2012 8th International Conference on Computing Technology and Information Management (NCM and ICNIT), Seoul, pp. 561–567 (2012)
28. Zainol, A., Mansoor, S.: An investigation of a requirements management tool elements. In: 2011 IEEE Conference on Open Systems (ICOS), pp. 53–58 (2011)
29. Niazi, M., El-Attar, M., Usman, M., Ikram, N.: GlobReq: a framework for improving requirements engineering in global software development projects: preliminary results. In: 16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012), Ciudad Real, pp. 166–170 (2012)
30. Ernst, N.A., Murphy, G.C.: Case studies in just-in-time requirements analysis. In: 2012 Second IEEE International Workshop on Empirical Requirements Engineering (EmpiRE), Chicago, IL, pp. 25–32 (2012)
31. Kumar, M., Shukla, M., Argarwal, S.: A hybrid approach of requirement engineering in agile software development. In: 2013 International Conference on Machine Intelligence and Research Advancement (ICMIRA), pp. 515–519 (2013)
32. Eberlein, A., Leite, J.: Agile requirements definition: a view from requirements engineering. In: Proceedings of the International Workshop on Time-Constrained Requirements Engineering (TCRE 2002), pp. 4–8 (2002)
33. Loconsole, A.: Empirical studies on requirement management measures. In: Proceedings of the 26th International Conference on Software Engineering, pp. 42–44 (2004)
34. Dube, R.R., Dixit, S.K.: Process-oriented complete requirement engineering cycle for generic projects. Presented at the Proceedings of the International Conference and Workshop on Emerging Trends in Technology, Mumbai, Maharashtra, India (2010)
35. Wnuk, K., Pfahl, D., Callele, D., Karlsson, E.: How can open source software development help requirements management gain the potential of open innovation: an exploratory study. In: Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, Lund, pp. 271–279 (2012)
36. Aurum, A., Wohlin, C.: A value-based approach in requirements engineering: explaining some of the fundamental concepts. In: Sawyer, P., Paech, B., Heymans, P. (eds.) REFSQ 2007. LNCS, vol. 4542, pp. 109–115. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73031-6_8
37. Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., Grünbacher, P. (eds.): Value-Based Software Engineering. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-29263-2
38. Siviy, J., Patrick, K., Lisa, M., John, M.: The value of harmonization multiple improvement technologies: a process improvement professional's view, Software Engineering Institute (SEI), Carnegie Mellon University (2008)
39. Urs, et al.: A unified process improvement approach for multi-model improvement environments (2009)
40. Ferreira, A., Machado, R.: Software process improvement in multimodel environments. In: 2009 Fourth International Conference on Software Engineering Advances. IEEE (2009)
41. Gasca-Hurtado, G.P., Hincapié, J.A., Muñoz, M.: Software process improvement assessment for multimodel environment tool to diagnose an organization. In: 2017 12th Iberian Conference on Information Systems and Technologies (CISTI). IEEE (2017)
42. Cross, N.: Designerly ways of knowing: design discipline versus design science. Des. Issues **17**, 49–55 (2001)

43. Hevner, A., Chatterjee, S.: Design science research in information systems. In: Hevner, A., Chatterjee, S. (eds.) Design Research in Information Systems, vol. 22, pp. 9–22. Springer, Boston (2010). https://doi.org/10.1007/978-1-4419-5653-8_2
44. Gasca-Hurtado, G.P., Muñoz, M., Mejia, J., Calvo-Manzano J.A.: Software requirements development: a path for improving software quality. In: Barafort, B., O'Connor, Rory V., Poth, A., Messnarz, R. (eds.) EuroSPI 2014. CCIS, vol. 425, pp. 194–205. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43896-1_17