2013 International Conference on Software and Systems Process (ICSSP)

Proceedings

Jürgen Münch,
Jo Ann Lan, and He Zhang

May 18–19, 2013
San Francisco, CA, USA

The Association for Computing Machinery, Inc.
2 Penn Plaza, Suite 701
New York, NY 10121-0701

Notice to Past Authors of ACM-Published Articles

ACM intends to create a complete electronic archive of all articles and/or other material previously published by ACM. If you have written a work that was previously published by ACM in any journal or conference proceedings prior to 1978, or any SIG Newsletter at any time, and you do NOT want this work to appear in the ACM Digital Library, please inform permissions@acm.org, stating the title of the work, the author(s), and where and when published.

ACM ISBN: 978-1-4503-2062-7

Additional copies may be ordered prepaid from:

ACM Order Department
P.O. BOX 11405
Church Street Station
New York, NY 10286-1405

Phone: 1-800-342-6626
(U.S.A. and Canada)
+1-212-626-0500
(All other countries)
Fax: +1-212-944-1318
E-mail: acmhelp@acm.org

Production: Conference Publishing Consulting, D-94034 Passau, Germany, info@conference-publishing.com

Process Variability Management in Global Software Development: A Case Study

Tomás Martínez-Ruiz, Félix García, Mario Piattini, Francisco de Lucas-Consuegra
Alarcos Research Group, Institute of Information Technologies and Systems, University of Castilla-La Mancha
Camino de Moledores, s/n, 13051 Ciudad Real, Spain telf. (+34) 926 29 53 00 ext. 96678
{tomas.mrtnez, delucasfrancisco}@gmail.com, {Felix.Garcia, Mario.Piattini}@uclm.es

ABSTRACT
Global Software Development (GSD) is set to be the paradigm that will support software industries in the increasingly globalized 21st century. It opens the door to companies from emerging countries to compete for their own gap in the market. It does, however, still bring some challenges with it. It must integrate different cultures, work styles, and work timetables in the same development process. In fact, GSD methodologies do indeed include specific activities to coordinate different work teams, but they fail precisely where any other methodology does: in the need to be truly useful by meeting the distinct cultural requirements of every organization involved, all at the same time. Up to now, process tailoring has been managed through variability mechanisms. Since these successfully merge original structure with cultural assets, they are also useful for adjusting global methodologies so that they suit each particular development context. This paper presents a case study of the use of the Variant-Rich Process paradigm (VRP) to support tailoring in a GSD methodology. It reveals the suitability of the VRP mechanisms, given that they support the two tailoring dimensions a GSD project involves, i.e., they take into account the circumstances of the entire global project, as well as the need to fit the internal characteristics of each organization; furthermore, they save effort in the tailoring process.

Categories and Subject Descriptors
K.6.3 **[Management of Computing and Information Systems]:** Software Management – *software process, software development, software maintenance.*

General Terms
Management, Documentation, Experimentation

Keywords
Process tailoring; Global Software Development; Variant-Rich Process paradigm; Process institutionalization

*ICSSP'*13, May 18–19, 2013, San Francisco, CA, USA

1. INTRODUCTION
Traditionally, it has been commonly accepted that software processes must fit the organization, because if not, the work force will reject them sooner or later [1]. The first consequence of this is that these processes are tailored before each enactment [2]. Literature therefore shows that the process tailoring problem has been dealt with by means of applying assets from products. Approaches based on components [3], product lines [4], even aspects [5] have been used in managing variability with traditional software engineering projects.

At the present time globalization also influences software engineering; this in turn brings about Global Software Development (GSD) projects. Globalization provides software development with several advantages; in addition to the possibility of saving resources or reducing the time to market, it makes it possible for the most qualified human resources to work together. Several pieces of work have already studied the benefits of this: cost-saving when accessing large multi-skilled workforces, reduced time to market, increased working-days, up to almost 24 hours [6-10]. Some challenges still remain, such as access to innovation and shared best practices and knowledge, improvement of resource allocation as well as of task modularization and communication. There is also a need for a clearer definition of the common process(es) [6, 11, 12] that coordinate(s) and improve(s) the work across all the organizations involved in a GSD project.

Even when the methodology is defined, GSD projects mean an extra challenge: as there are several organizations involved in the same project, the processes must meet the requirements of all these organizations as they work together. As each one possesses its own particularities, the same processes must fit all of the organizations at one and the same time. It is true that using a methodology mitigates the problems of GSD, but the real benefits are clearly unattainable, and the advantages of GSD are, moreover, placed under threat unless the methodologies are able to be set in motion and run in different cultures. Thus suitable variability support counts towards the usefulness of GSD.

Unlike traditional projects, tailoring in global contexts must be managed from a multidimensional viewpoint, one dimension or even more per organization involved. To support process tailoring in the GSD scenario, traditional approaches must be enhanced and ad hoc tailoring becomes unacceptable, given the special complexity of global development. Under those circumstances, the enactment of GSD methodologies requires mature process tailoring approaches, such as product lines and aspects-based ones.

In previous work we have defined a process institutionalization framework, which includes the Variant-Rich Process paradigm

[13]. This has been designed to provide software industries with the process tailoring support they actually require [14]. The paradigm and the notation implementing it, vSPEM, have been tested in several experiments and case studies, moreover [15, 16]. They have been shown to be extremely applicable in the support of process tailoring. This work now centers on applying our paradigm to model the variability in the case of a GSD methodology in the context of the ORIGIN project [17]. The aim is to turn the ORIGIN methodology into a variant-rich process, which could easily be tailored to any organization involved in a GSD project. We also present the vEPF plugin, which provides modeling support to vSPEM.

This paper is structured as follows. Section 2 describes the state of the art. Our SPRINTT framework, and the Variant-Rich Process paradigm are presented in Section 3, along with the vSPEM notation. Section 4 deals with the case study. The plugin giving technical support to the vSPEM language is introduced in Section 5. Finally, Section 6 outlines our conclusions and future work.

2. STATE OF THE ART

Variability as a support for process tailoring has been widely dealt with in literature. Processes had been tailored by means of changes made to their structures [18, 19]. After that, some pieces of work concerning this theme are the "process lines" approach proposed by Rombach [4]; and the links between aspects and processes, by Sutton [5]. The systematic literature review presented in [14] describes how processes are modified in order for them to fit the project's needs; it also states the set of requirements a process variability notation must include to support tailoring as real organizations actually need it. It therefore guides the definition of new process variability support mechanisms.

Some other new work has also appeared since the aforementioned systematic review. First of all, the article by Simidchieva *et al.* [20] presents an explicit differentiation between problem and solution spaces, and identifies three types of approaches: generation, navigation and reasoning. Araujo *et al.* [21] propose the management of process variability by identifying the common, mandatory, optative and alternative features of a process model. The same authors have also defined a tool that imports method plugins from EPF also proposing the definition of a nine-step procedure with which to apply MDE transformations to process tailoring [21]. Simmonds *et al.* [22] propose the creation of Basic Feature Models to represent features of the tailored process over the vSPEM notation (which is set out in Section 3), and they also pay special attention to orthogonal (crosscutting) variations. Hurtado Alegría *et al.* [23] propose tailoring software processes by using MDE and ATL transformations to convert generic variable processes into specific tailored ones.

Of those initiatives which apply variability to software processes in order to align the processes themselves with the projects found, we should highlight the work of Martins and Silva [24, 25], a proposal based on three fundamental steps: i) defining the process, ii) adapting and monitoring the process execution, and iii) measuring the process. Killisperger et al, [26] suggest an environment through which to apply variations to processes automatically through variation operations. Silva Barreto et al, [27, 28] propose another environment in which to carry out variations in software processes, with the aim of facilitating process reuse, based on the definition of variations in process components.

Table 1 summarizes the main characteristics of the process improvement frameworks. All of them include some aspects that seek continuous process improvement or institutionalization (as is defined in this article), but none of them really support process tailoring which serves to provide processes that fit their organizations. It should also be said that, to the best of our knowledge, the tailoring approaches identified do not (at all) meet the requirements industries require, as has been pointed out in the SLR of [14], a fact that will be observable in Table 2. Bearing all the above in mind, as well as our goal of supporting process institutionalization systematically by means of adaptation and standardization, a framework with a variability modeling notation has been created, which fulfills the requirements that have been set out.

Table 1. Characteristics of the most-related work

Characteristics	Martins *et al.*, 2009	Killispeguer *et al.*, 2009	Silva *et al.*, 2008, 2011
Based on process tailoring (to project)	YES	YES	YES
Focused on process improvement	YES	NO	NO
Including cyclic initiative	NO	NO	NO
Including different stages (traceable)	YES	YES	YES
Standardizing processes at organizational level	NO	NO	YES
Knowledge use and reuse	YES	NO	YES

Table 2. Requirements fulfillment by existing approaches

Requirement	Simidchieva *et al.*, 2012	Araujo *et al.*, 2011	Simmonds *et al.*, 2011	Hurtado *et al.*, 2011
RQ1. Variation support in the process composing elements	7/8	3/8	5/8	6/8
RQ2. Types of supported variation	3/10	7/10	7/10	7/10
RQ3. Notation supporting tailoring	1/2	1/2	1/2	1/2
RQ4. Tailoring support	2/5	3/5	1/5	2/5

3. THE SPRINTT APPROACH

The *Software Process Institutionalization based on Tailoring and sTandardization* (SPRINTT) approach is composed of two elements, which have been termed the *Institutionalization Cycle* and the *Variant-Rich Process paradigm* (VRP). The former is the theoretical setting through which organizations transform and include processes as new and effective assets. It defines four cyclical steps to *tailor*, *execute*, *analyze*, and *standardize* the processes, just as Figure 1 shows.

The Variant-Rich Process paradigm offers variability support which is suitable for the execution of both the first and last steps

of the cycle. The paradigm provides on-point and crosscutting variability mechanisms, as well as Rationale management to give support during the decision support. These are based, respectively, on Product Lines (SPLE's) [29], Aspects (AOSE) [30] and Rationale Management [31] from Software Engineering.

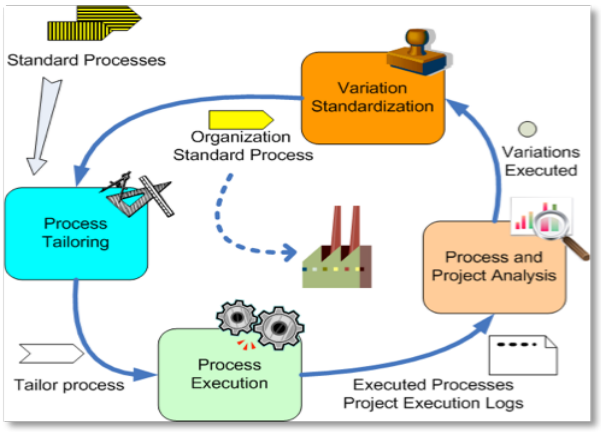

Figure 1. Institutionalization Cycle overview

The VRP gives tailoring support as industries require, just as the systematic review illustrated in summary form [14]. This SLR also stated that variability is needed in different process notations. Consequently, the VRP includes generic variability mechanisms. These have been implemented over SPEM [32], resulting in the vSPEM language [15, 16, 33-36]. By using the vSPEM notation it is possible to define variant-rich processes in a diagram (Figure 2), which is tailored by using the on-point and crosscutting variations (Table 3).

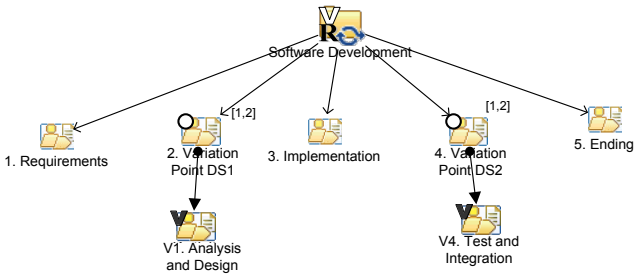

Figure 2. A variant-rich process including some on-point variations

Table 3. Grammar for modeling a crosscutting variation [34]

Aspect	Aspect Variation2{

Process pointcut	pointcut ppc1 (VPTask vpt1, VPWorkP vpw1,){
	vpt1=(execution("1.2.2*"));
	vpw1=(produce(*)&& within("1.2.2*"));
	}

Process advice	advice ppc1 (VPTask vpt1, VPWorkP vpw1){
	vpt1.occupe(Analyse HW SW Interaction);
	vpw1.occupe(Software Design); }
	}

The VRP paradigm has been tested using the vSPEM notation. Experiments have shown it is much easier to adapt processes if they are modeled using the vSPEM language, even considering the lack of understanding of a new notation (and paradigm) [16]. On the other hand, these results have clearly demonstrated that the VRP and the notation are useful in modeling variability in Aerospatiale and Quality Assurance domains [15, 34]. This article is now focused on presenting a new case study using a GSD methodology, and the tool that support process variations.

4. CASE STUDY. ORIGIN

The case study [37, 38] has focused on checking if the Variant-Rich Process paradigm is suitable (useful and practical) for modeling variations in a real Global Software Development methodology through the use of the vSPEM.

The object of study has been both the new paradigm and the notation, used to model variability of the ORIGIN methodology. The methodology was developed in the context of the ORIGIN project, which focused on supporting global software development.

ORIGIN [17] includes the guidelines for the management and development of global projects. Based on PUD [39] SCRUM [40], it also includes Agile practices [41, 42], to foster a greater sense of satisfaction in, and with, their own work on the part of the participants . ORIGIN includes assets such as methods with which to elicit requirements, ensure quality, execute tests, and plan projects and their management in GSD projects. It supports different team structures and physical locations, promotes relationships between partners, while at the same time improving interaction and coordination between workers. All of the projects follow different standards or can be executed in different ways, depending on their particular type, or on other factors. In short, ORIGIN has been designed to provide full support to GSD projects, but as with any methodology, it must be tailored to fit each organization's needs, if is to be useful in reality.

4.1 Subjects and Analysis Units

The processes used in the case study are included in the ORGIN GSD methodology. This is the main asset from the ORIGIN project, which has involved various enterprises and the Alarcos Research Group from the University of Castilla-la Mancha. The processes were initially modeled in a non-variability support language, and the variability needs were therefore implicitly included in the textual definition of the processes. As a result, the variant-rich version of the ORIGIN process has been created from scratch, i.e., from variability textual requirements.

4.2 Variability Planning

Modeling variability in ORIGIN was executed by following a procedure based on experience from former case studies (Table 4). This procedure aims to create the ORIGIN variant-rich process, from the textual analysis of the variability requirements included in the methodology. The following sections summarize the main results obtained by following this procedure.

Table 4. Procedure to execute the case study

1)	Context Analyses of Variabilities
	a) Context scoping
	b) Syntactic Analysis
	c) Semantic Analysis
2)	Design of Variations:
	a) Definition of variations.
	b) Search for and stating of new variations
3)	Implementation of Variations
	a) Implementation of on-point variations
	b) Implementation of crosscutting variations
4)	Presenting the resulting Variant-Rich Process

4.3 Tailoring Requirements from the Context Analysis

The ORIGIN methodology includes several needs for adaptation to meet the actual context of each enterprise. After executing the context analysis, these requirements have been elicited and classified, along with the factors influencing its tailoring.

- *Time Difference*: Time differences affect the communication and interrelation between the distributed teams around the world; thus the entire development of the project is affected. Some cultural aspects mean that working time does not always correspond with clock time (working times in Spain and Germany are different, even though they share the same time zone). It should also be remembered that the best use of the working day is more or less around the middle of it, so meetings tend to be planned at those times. These differences must therefore be considered in terms of the level of overlapping between the working times of the different teams. This level ranges form a total overlap, in which the work teams can communicate with each other by synchronous method, to null overlap. This latter term means that the teams work at different times, so must use asynchronous communication methods.
- *Distributed Meetings*: It is clear that if the time overlap is low, it is difficult to plan meetings to coordinate people. This being so, the problem of a lack of meetings must be solved by means of documentation so all the teams can follow and understand the project without any problem. Where a high level of overlapping exists, videoconferences may be used instead. According to Woodward [43] there are five methods to carry out distributed meetings that may be used, *documentation meetings*, *link approach*, *altering schedule*, *sharing the grief*, and *feel the grief*.
- *Creating Distributed Teams* (from a Characteristic Team) is one of the first steps to be carried out. The teams – known as characteristic teams - must now assume different roles (analysts, designers, programmers), and all of them must be capable of developing one characteristic from the Product Backlog. It is recommended that the members of the same team come from the same locations, but there are certain ways to convert the component teams into characteristic teams. Larman and Vodde propose *big-bang reorganization*, *gradual expansion of responsibility*, and *gradual introduction of characteristic teams* [44].
- *Management of the Distributed Team*: It is a main factor in a distributed process. It affects several activities of the development phase and must be coordinated between all the companies. In fact, there are three methods: Isolated Scrums, Scrum of Distributed Scrums, and Integrated Scrums. The first is the most commonly-used, but it may be adjusted, since it affects meetings and certain tasks concerning the team.
- *Implementation and Testing Methodologies*: ORIGIN has been designed to follow the ATDD (Acceptance Test-Driven Development), and its use is recommended so that each work team may choose the methodology they are more confortable with. If necessary, different development and testing methodologies may be used.
- *Management of Shared Documents*: There are mechanisms with which to share documents that may be used in a distributed project; by default, the use of a wiki is by far the most widely-recommended. In addition, different constraints regarding communication may make one of the mechanisms more useful than the others.
- *Retrospective of Retrospectives*: This activity is planned to be executed in the Spin Retrospective phase. This activity consists of a meeting to analyze the dependencies between the different development groups, and analyses the sprint from the coordination viewpoint. As its definition suggests, this activity is not necessary in cases in which there are no dependencies between the teams.
- *Team Size*: The work team is the multifunctional people group needed to execute the work and to convert the product backlog into a functional product. According to ORIGIN, its size must be of 7±2 people. However, sometimes certain roles, such as the Product Owner and Scrum master, may be included in the work team.
- *Scrum of Scrums Meeting*: The scrum of scrums meetings activity focuses on the weekly coordination of the different teams, in order to look at the dependencies, problems and so on. Different representatives of each team may take part in the meeting or, if there are no dependencies between the teams, it does not take place.

Most of these tailoring requirements affect how the entire GSD project is set up. They come about mainly as a result of the real state of communication, and must be fixed or decided within the scope of the whole project. All the organizations involved in the GDS project must reach an agreement about these, and tailor the ORIGIN methodology accordingly.

There are some other variations, however, that do not affect all the organizations as a whole, but only influence how each organization implements the project at a local level. In these cases agreement between the organizations is not required, so each one of them may tailor the ORIGIN processes according to their own particular needs, as well as to their own culture.

Table 5 summarizes the scope of the elicited variations. The ORIGIN processes must therefore be tailored according to the requirements of all the organizations involved; these together make up one project dimension. All the specific dimensions, including the characteristics of each organization, must also be taken into account in the tailoring. Variability mechanisms must support these dimensions, while also tailoring in different times.

Table 5. Scope of the elicited variability requirements

Variation	Scope
Distributed Meetings (time difference)	Project
Creating Distributed Teams (from a Characteristic Team)	Project
Management of the Distributed Team	Project
Implementation and Testing Methodologies	Organization
Management of Shared Documents	Project
Retrospective of Retrospectives	Project
Team Size	Organization
Scrum of Scrums Meetings	Project

4.4 Definition of the ORIGIN Variant-Rich Process

After considering the tailoring requirements ORIGIN includes, the ORIGIN Variant-Rich Processes have been outlined, first of all, by means of defining the variability elements giving support to these variations. Tables 6 and 7 summarize the *on-point* and *crosscutting* variability elements, respectively.

Table 6. Presentation of the *on-point* variability elements

Variation	Type	Var. Points	Variants
Implementation and Testing Methodologies	Alternative.	2 variation points	2n variants[1]
Retrospective of Retrospectives	Optional	One variation point	One variant
Team Size	Alternative	One variation point	n variants[1]
Scrum of Scrums	Optional	One variation point	One variant

Table 7. Presentation of the *crosscutting* variability elements

Variation	Aspects	Variants in the Aspect	Variation Points Affected[2]
Distributed Meetings (time difference)	5 aspects (one per method described)	5n variants[3]	n variation points
Creating Distributed Teams (from a Characteristic Team)	3 aspects (one per mode of creating the team)	3n variants[3]	n variation points
Management of the Distributed Team	3 aspects (one per mode of management)	3n variants[3]	n variation points
Management of Shared Documents	3 aspects (one per mode of sharing documents)	5n variants[3]	n variation points

4.5 Implementation of the ORIGIN Variant-Rich Process

The *on-point* and *crosscutting* variabilities have been modeled, by using the mechanisms proposed in the VRP.

4.5.1 Implementation of the On-Point Variations

On-point variations are implemented through the creation of variation points, variants, and the variability dependencies between these. They are: empty points in the process structure to execute variability; specific implementations of this variability; and the rules forcing consistence during the process tailoring. To illustrate these, the optional on-point variation as regards the *retrospective of retrospectives* activity is presented.

In the first place, an activity variation point is placed in the corresponding *Sprint Retrospective* phase (instead of the original activity). Figure 3 presents the variation point.

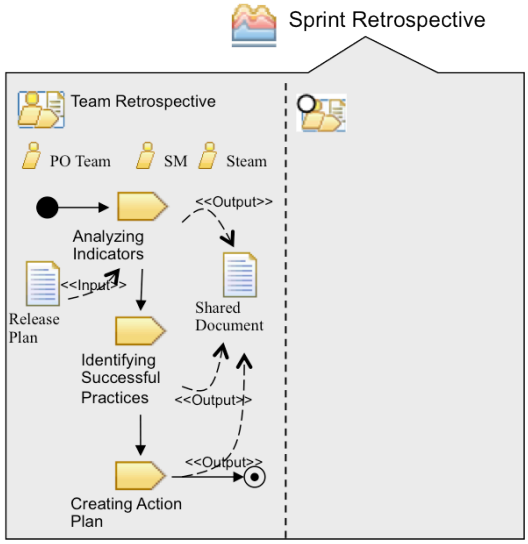

Figure 3. Variation point inside the Sprint Retrospective phase

Additionally, an activity variant implementing the *retrospective of retrospectives* is specifically designed for placing in the previous variation point. Figure 4 presents its definition in diagram form.

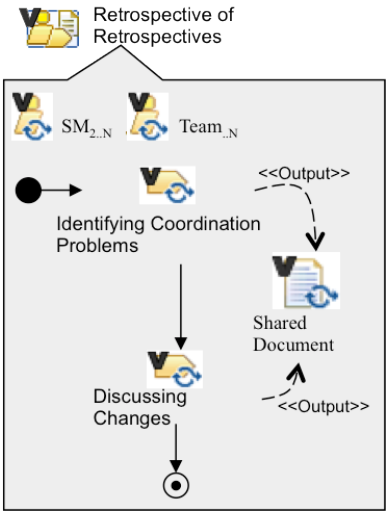

Figure 4. Variant of the retrospective of retrospectives activity

Finally, tailoring consistence is also embedded into the variant-rich process by means of dependencies (*variant2variation* point and *variationPoint2variant*).

4.5.2 Implementation of the Crosscutting Variations

By definition, crosscutting variations execute a lot of on-point variations. They are composed of one or more aspect; each one of these tailors the process in one particular way. Aspects have the capability of executing several on-point variations at once. They analyze the entire process structure and locate the specific points

[1] There are as many variants as possible implementations of the variability

[2] Variation points are not explicitly designed. They will be retrieved by process pointcuts. This column is shown as illustration

[3] A variant is defined per each aspect, and per each point the crosscutting variation affects

(by means of the pointcuts) in which to execute these variations, using the advices.

Table 8 describes the *DocumentationMeetings* aspect (an overview), which is affected by the time difference.

Table 8. Description of the aspect *documentationMeetings*

Aspect	Aspect *documentationMeetings*{
Pointcuts	Process pointcut *activitiesAffected* (vpActivity vpa_a1, vpActivity vpa_a2…){ Vpa_a1::=(executing "weekly scrum"); Vpa_a2::=(executing "scrum of scrums); } …. //other pointcuts
Advices	Advice *activitiesAffected* (vpActivity vpa_a1, vpActivity vpa_a2…){ Vpa_a1.free(); Vpa_a1.occupe("weekly_scrum_documentation"); Vpa_a2.free(); Vpa_a2.occupe("scrum_of_scrums_document."); } …//other advices
	}

The aspect shown in Table 8 tailors the methodology according to the *DocumentationMeetings* mode. The first part (pointcuts) looks for specific meeting activities throughout the whole methodology, and obtains the exact points of the process structure in which documentation meeting activities must be inserted. Then, the second part (advices) places the correct meeting activities in these holes.

4.6 Presenting the Resulting ORIGIN Variant-Rich Process

The resulting ORIGIN Variant-Rich Process has been presented to the developers of ORIGIN, in order to obtain their agreement. The usage of variability mechanisms has also been described.

4.6.1 On-Point Variation Retrospective of Retrospectives

The activity *Retrospective of Retrospectives* may or not be included in the corresponding *Sprint Retrospective* phase. If it is not included, there is no tailoring element required, and then the variability elements disappear when variability is interpreted. The tailored process is presented in Figure 5.

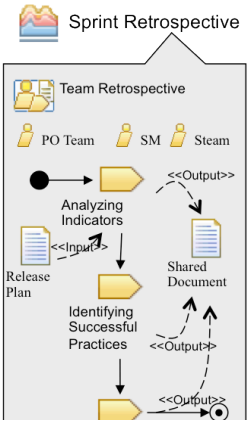

Figure 5. Tailored phase without using the R*etrospective of Retrospectives* activity (overview)

If, on the other hand, this activity is included, a variation is now created by means of using the variability mechanisms defined. Figure 6 shows this in a diagram. When variability is interpreted, all the occupation relationships turn the variants and variation points they link into specific process elements. Regarding this on-point variation, in a concrete activity (Figure 7).

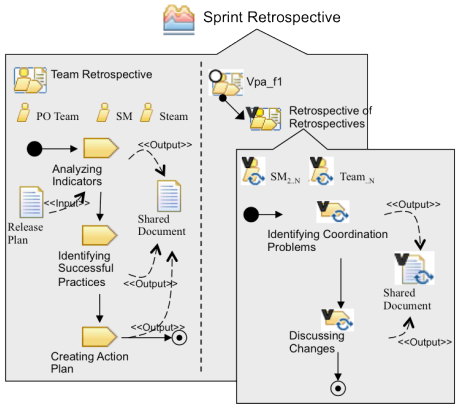

Figure 6. Variation about the *Retrospective of Retrospectives*

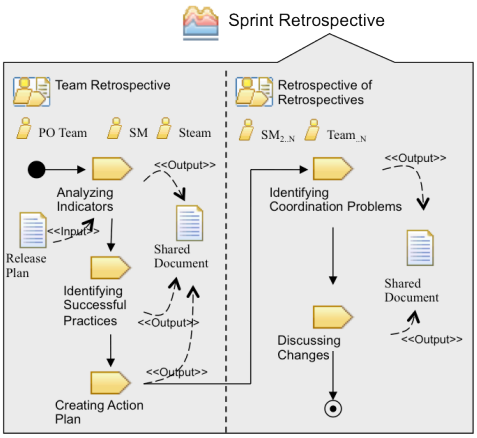

Figure 7. Interpretation of the on-point variation

4.6.2 Crosscutting Variation about Documentation Meetings

Meetings in a GSD methodology clearly depend on the time overlapping between the companies involved. In this case, the ORIGIN has been tailored in a case where there is no time overlapping, and companies need to communicate with each other by means of the documentation asynchronous mode. The *DocumentationMeetings* aspect is therefore activated. Firstly, it will automatically filter (using the pointcuts) the process-composing elements affected by the communication mode. Figure 8 goes on to show two activities affected by the type of meetings, which have been transformed into empty points.

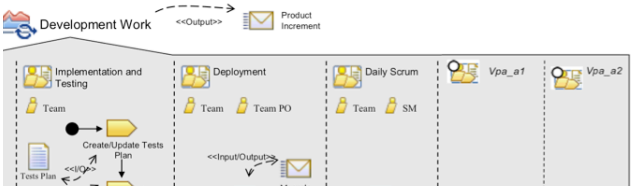

Figure 8. Variation points filtered by the process pointcuts (overview)

Once the points affected by the type of meetings are localized, they are used to automatically place the activities implementing the selected type of meetings. Figure 9 presents the execution of two of these variations. Figure 10 shows the final phase after the entire process is tailored.

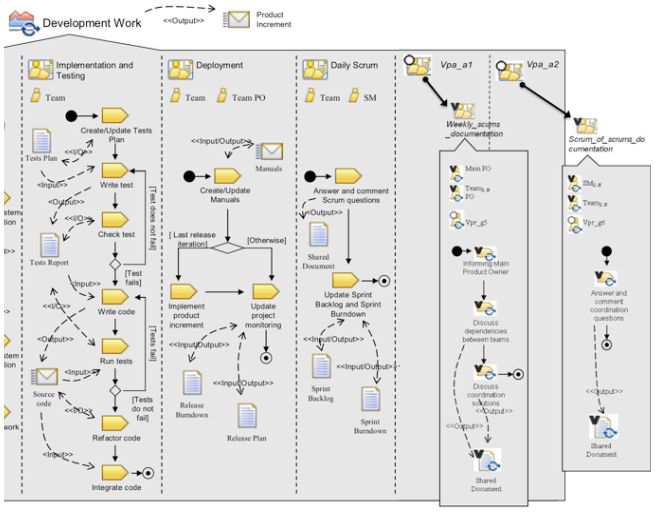

Figure 9. On-point variations executed by the advices

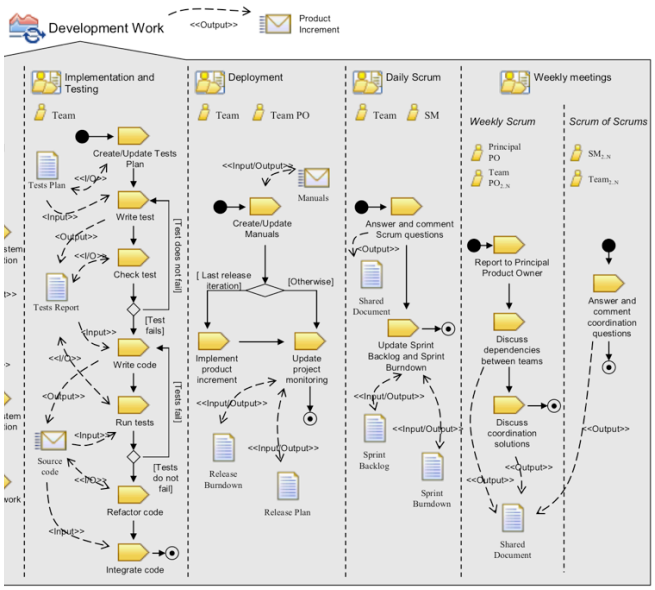

Figure 10. Final tailored activity

4.7 Validity and Limitations

Threats to validity have been dealt with as regards the construct, internal and external validities [38]. Regarding construct validity, the research question focused clearly on attempting to apply the paradigm and the notation to real variations in GSD processes, in order to check its suitability.

In this study, one of the main factors affecting the internal validity was that the designer of the VRP executed the case study. The fact that he had previous knowledge of the variability mechanisms would have some affect. However, since the case study did not focus on assessing the efficiency but on whether these mechanisms could really be applied to industrial processes, this threat does not affect the results. The process model and the variation needs were taken from a real enterprise focusing on Global Software Development, and the results obtained were therefore dependent on that sector of the market. However, previous replicas of the study were carried in an effort to minimize risks about external validity.

Since the objective of the study was not related to efficiency, the results are not dependent on the experience of the researcher; he thus does not affect the reliability. Any person who has had more or less experience of using the notation or the paradigm and who has the time needed will be capable of modeling the variability presented in ORIGIN.

4.8 Lessons Learned

Several lessons are learnt from the execution of this case study.

- First of all, GSD methodologies include variability, as every process model does. Sometimes processes make explicit reference to variability or tailoring support (or adjustment mechanisms), but in most cases it is implicitly defined. This case study shows the need to tailor support by using variability mechanisms even more clearly: GSD processes structure the work that all the organizations involved in a GSD project do, so it is important for these processes to satisfy those needs so that, organizations can implement the processes.
- Processes must be tailored to meet requirements from several different viewpoints at the same time. On one hand, they must fit the context in which all the organizations are working together, as well as the specific requirements each organization internally needs to satisfy in its work. From a more abstract viewpoint, GSD methodologies bring up the challenge of tailoring in a general dimension, and after that, in as many particular dimensions as there are firms involved in the GSD project.
- Regarding the objective of the case study, it has been fulfilled. It must be highlighted that the vSPEM notation (and hence, the VRP paradigm) is fully applicable in modeling variability and providing tailoring support in Global Software Development methodologies. In fact, the Variant-Rich Process paradigm provides the mechanisms for tailoring the GSD processes in the general, and all the particular, dimensions any project involves.
- Moreover, extracting variability from the textual description of the processes and modeling it requires a well-defined and well-structured procedure, so as not to build in bias or inconsistencies, and to build variant-rich processes with reliable tailoring support.
- Finally, a post-mortem analysis of tailoring effort has been executed. It compares the investment and cost of using the VRP tailoring mechanisms and the "traditional" ones. Using the quantitative results from the experiments, it is possible to infer that the initial effort in creating the ORIGIN variant-rich process – most of which is due to the knowledge discovering- means a nearly constant effort in defining each tailored process, while traditional approaches mean a lineal cost in tailoring new processes, as Figure 10 presents.

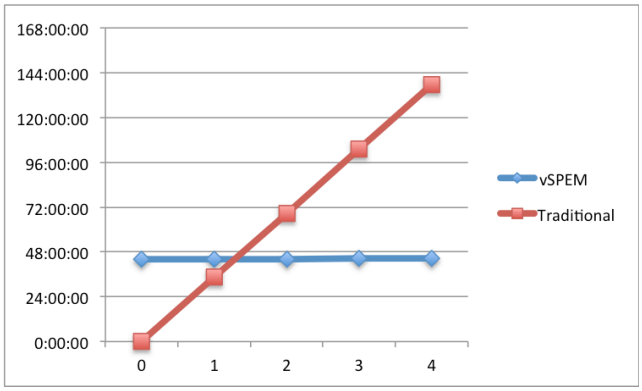

Figure 11. Comparison of tailoring costs

5. VEPF TOOL

To give efficient support to the variability mechanisms included in the vSPEM notation, the previously existing EPF Eclipse plugin [45], has been improved through vEPF. The first version of vEPF4 (vEPF 1.0) implements the on-point variability mechanisms. It was created in two phases; the first included a full reverse engineering over EPF that provided knowledge in detail about how the plugin runs, as well as how to modify and improve it, while the second one was divided into several iterations, each of which focused on adding well-defined functionality to several process-composing elements. The plugin has been created to provide full compatibility with EFP processes.

The functionality of vEPF is clearly divided into two subsections. On the one hand, it provides mechanisms with which to create variant-rich processes (commonly known as process lines), and on the other, it includes the assets needed to tailor processes from them.

The next version of vEPF is being developed. It will include support to crosscutting variations, as well as knowledge management.

5.1 Defining Variant-Rich Processes

The vEPF Library has been enhanced to include variant-rich processes (VRPs) (Figure 12). VRPs are created from methods (just as EPF supports), but these include some variability mechanisms that must be tailored towards concrete processes later.

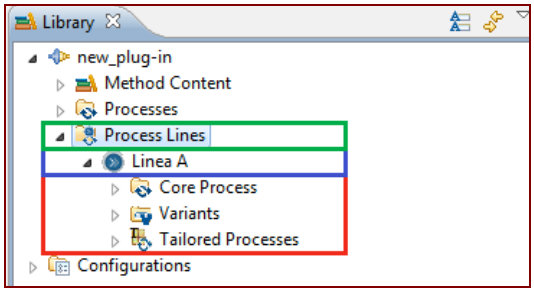

Figure 12. Overview of the vEPF library

Variation points and variants are included in the variant-rich process by using the New Child option (just as other process elements do). Their properties, and dependencies with other variability elements, are set by means of a specific form, as Figure 13 shows in the case of variation points. A similar one has been created for variants. Variation points are fixed in the process breakdown structure; variants are included as "floating" elements, whilst they are available to tailor software processes (Figure 14).

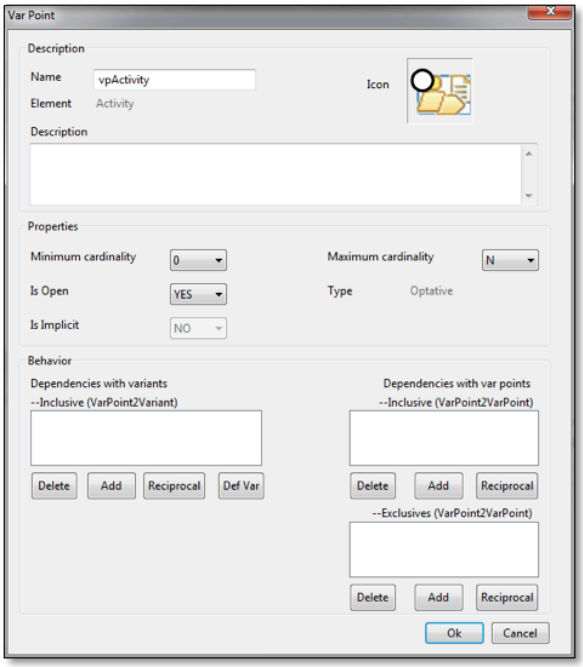

Figure 13. Way of defining variation points

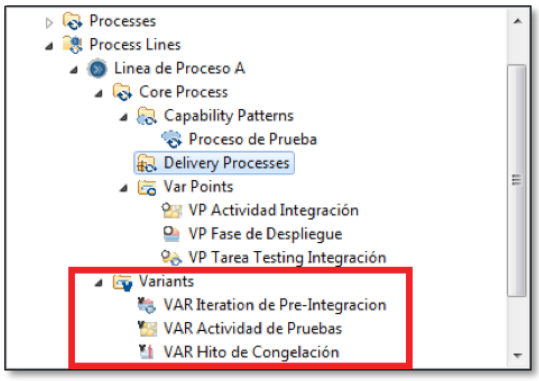

Figure 14. Variants to tailor the process

5.2 Tailoring processes with vEPF

Once the variant-rich processes (namely process lines) have been defined, they can be used to tailor new processes. vEPF also supports the adaptation of new processes.

In this case, the process-tailoring window appears (Figure 15), supporting the definition of occupation relation-ships, namely, the inclusion of variants in the variation points by means of the specific variations window. It also controls that the dependencies between all the variability elements are satisfied, before tailoring is completed.

Once all the mandatory variation points have been filled with a variant, the process is tailored. Figure 16 presents a tailored process which summarizes all the variations executed and the elements that they affect

[4] http://alarcos.esi.uclm.es/vepf

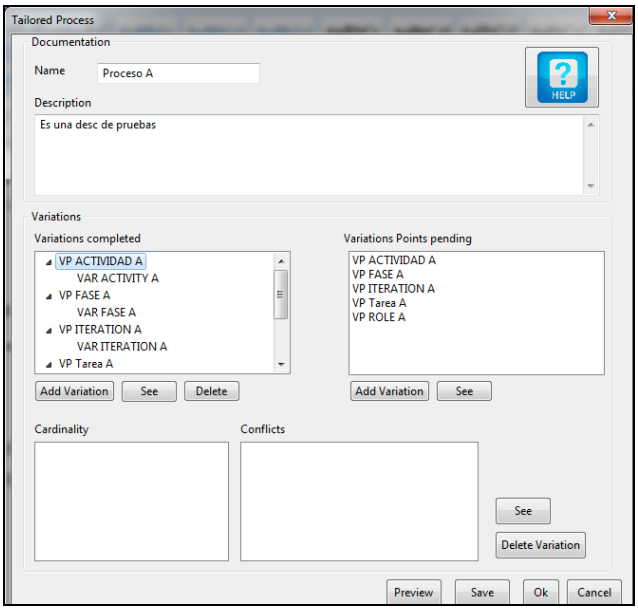

Figure 15. Process tailoring

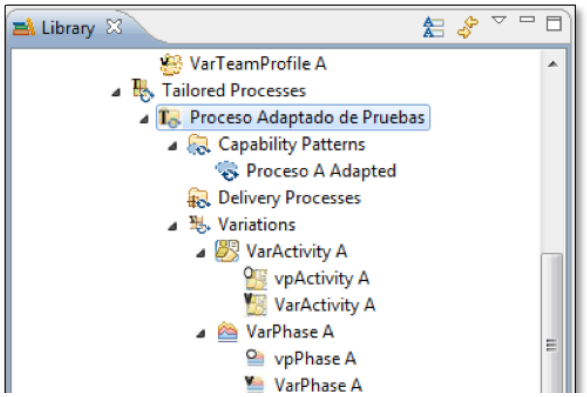

Figure 16. Tailored process with variants in all the variation points

6. CONCLUSIONS

Enactment of traditional processes requires suitable mechanisms, such as the VRP proposes. They provide these processes to be set in motion in the organization performing them. When the number of organizations involved in the project increases, as happens with GSD projects, the variability mechanisms remain fundamental.

GSD processes require tailoring from general and individual perspectives. Firstly, the processes must fit in to the context of all these organizations as a whole, as well as the project itself. After that, this shared process must be undertaken by each organization individually. The culture, the working style, the characteristics of its working teams set up a dimension of each organization that the process must be compatible for. This means that process tailoring in a GSD context must make different process instances compatible in a multidimensional context, and variability mechanisms should have the capability of managing them. The case study presented in this article clearly sets out the variabilities the ORIGIN methodology implies, through a detailed textual analysis, right through to a complete description. It shows that some of them depend on the GSD project, and on how the whole organizations work together. Some others, such as the size of the teams, are dependent on each organization.

The case study has also shown that the variability mechanisms of the Variant-Rich Process paradigm, the vSPEM notation implements, are suitable (useful and practical) in modeling the variabilities included in a GSD methodology. In fact, they give full support to tailoring GSD processes from the project and organizational perspectives, which means that this kind of tailoring technology also remains fundamental in GSD process enactment. The effort in tailoring ORIGIN from scratch and from the ORIGIN variant-rich process (considering the effort in creating it) has been also compared, taking into account the real effort per variation from previous experiments. These results clearly show that the Variant-Rich Process paradigm provides cheaper tailoring support.

The vEPF tool has been created with the aim of bringing the support that process tailoring really needs. It has been designed to automate, as well as manage tailoring knowledge by using the vSPEM language. The current version supports on-point variations, while the next version is planned to give full support to the VRP.

Future work will deal with the issues in different ways. First of all, new case studies are planned, to check the usability of the VRP in several domains, as well as the implementation of the SPRINTT cycle over these organizations, seeking to validate it and to obtain feedback from the organizations. Another aim includes completing the second version of the vEPF tool, to integrate crosscutting variability mechanisms giving full support to GSD process tailoring and enactment.

7. ACKNOWLEDGMENTS

This work has been funded by the GEODAS-BC project (Min. de Economía y Competitividad and Fondo Europeo de Desarrollo Regional FEDER, TIN2012-37493-C03-01).

8. REFERENCES

[1] Zahran, S. Software Process Improvement: Practical Guidelines for Business Success. Addison-Wesley, Harlow, United Kingdom, 1998.

[2] Yoon, I.-C., Min, S.-Y. and Bae, D.-H. Tailoring and Verifying Software Process. In Proc. of the Proc. of the 8th Asia-Pacific Software Engineering (Macao, China, December 2001, 2001). IEEE CS, Washington, DC, USA.

[3] Szyperski, C., Bosch, J. and Weck, W. Component-Oriented Programming. Springer-Verlag, City, 1998.

[4] Rombach, D. Integrated Software Process and Product Lines. Springer, City, 2005.

[5] Sutton, S. M. Aspect-Oriented Software Development and Software Process. Springer, City, 2005.

[6] Ebert, C. and Neve, P. D. Surviving Global Software Development. IEEE Software, 18, 2 2001), 62-69.

[7] Ågerfalk, P. J., Fitzgerald, B., Holmström, H. and Conchúir, E. Benefits of Global Software Development: The Known and Unknown. In Proc. of the International Conference on Software Process (2008). Springer Berlin / Heidelberg

[8] Humphrey, W. S. Introduction to the Team Software Process. Longman, Massachusset, USA, 2000.

[9] Cataldo, M. and Herbsleb, J. D. Communication networks in geographically distributed software development. In Proc. of the CSCW '08 Proceedings of the 2008 ACM conference on Computer supported cooperative work (2008)

[10] Damian, D. and Zowghi, D. The impact of stakeholders' geographical distribution on managing requirements in a multi-site organization. In Proc. of the Requirements Engineering (RE'02) (2002)

[11] Nguyen, T., Wolf, T. and Damian, D. Global Software Development and Delay: Does Distance Still Matter? In Proc. of the IEEE International Conference on Global Software Engineering (2008). IEEE Computer Society

[12] Solingen, R. V. and Valkema, M. The Impact of Number of Sites in a Follow the Sun Setting on the Actual and Perceived Working Speed and Accuracy: A Controlled Experiment. In Proc. of the 5th IEEE International Conference on Global Software Engineering (2010)

[13] Martínez-Ruiz, T., García, F. and Piattini, M. Process Institutionalization Using Software Process Lines. In Proc. of the 11th Int. Conf. on Enterprise Information Systems (Milan, 2009)

[14] Martínez-Ruiz, T., Münch, J., García, F. and Piattini, M. Requirements and Constructors for Modeling Variability in Software Processes, a Systematic Review. Software Quality Journal, 20, 1 2012), 229-260.

[15] Martínez-Ruiz, T., García, F. and Piattini, M. Supporting Software Process Variability: The Variant Rich Process Modelling Paradigm Sent to SOSYM2012).

[16] Martínez-Ruiz, T., García, F., Piattini, M. and Münch, J. Modeling Software Process Variability: An Empirical Study. IET Software, 5, 2 (March 2010 2011), 172-187

[17] del Nuevo, E., Piattini, M. and Pino, F. J. Scrum-based Methodology for Distributed Software Development. In Proc. of the ICGSE (2011)

[18] Sutton, S. and Osterweil, L. J. PDP: Programming a Programmable Design Process. In Proc. of the 8th Int. Workshop on Software Specification and Desing (Schloss Velen, 1996)

[19] Sutton, S. M. and Osterweil, L. J. Product Families and Process Families. IEEE CS, City, 1996.

[20] Simidchieva, B. I. and Osterweil, L. Categorizing and Modeling Variation in Families of Systems: a Position Paper. ACM, City, 2012.

[21] Araujo, F., Freire, M. A., Câmara dos Santos, W. and Kulesza, U. An Approach to Manage and Customize Variability in Software Processes. In Proc. of the 24th Brazilian Symposium on Software Engineeering SBES 2010 (Salvador, BA, 2010). IEEE Digital Library

[22] Simmonds, J. and Bastarrica, M. C. Modeling Variability in Software Process Lines. Departamento de Ciencias de la Computación. Universidad de Chile, 2011.

[23] Hurtado Alegría, J. A., Bastarrica, M. C., Quispe, A. and Ochoa, S. F. An MDE Approach to Software Process Tailoring. In Proc. of the Proceedings of the 2011 International Conference on Software and Systems Process (Waikiki, Honolulu, 2011). ACM, New York.

[24] Martins, P. V. and Silva, A. R. ProPAM: Discussion for a New SPI Approach. Software Quality Professional, 11, 2 2009), 4-17.

[25] Martins, P. V. and Silva, A. R. ProPAM. SPI based on Process and Project Alignment. IDEA Group Inc., City, 2007.

[26] Killisperger, P., Stumptner, M., Peters, G., Grossmann, G. and Stückl, T. Meta Model Based Architecture for Software Process Instantiation. Springer Verlag, City, 2009.

[27] Silva Barreto, A., Murta, L. and Rocha, A. R. Software Process Definition: a Reuse-Based Approach. City, 2008.

[28] Silva Barreto, A., Murta, L. and Rocha, A. R. Software Process Definition: a Reuse-Based Approach. Journal of Universal Computer Science, 17, 13 2011), 1765-1799.

[29] Clements, P. and Northrop, L. Software Product Lines. Practices and Patterns. Addison-Wesley., Boston, 2002.

[30] Filman, R. E., Elrad, T., Clarke, S. and Aksit, M. Aspect-Oriented Software Development. Addison-Wesley, Boston, MA, 2004.

[31] Dutoit, A. H., McCall, R., Mistrík, I. and Paech, B. Rationale Management in Software Engineering: Concepts and Techniques. Springer, City, 2006.

[32] OMG. Software Process Engineering Metamodel Specification. Object Management Group, 2008.

[33] Martínez-Ruiz, T., García, F. and Piattini, M. Managing Process Diversity by Applying Rationale Management in Variant Rich Processes. Springer Verlag, City, 2011.

[34] Martínez-Ruiz, T., García, F., Piattini, M. and Münch, J. Applying AOSE Concepts for Modeling Crosscutting Variability in Variant-Rich Processes. In Proc. of the 37th Conf. on Software Engineering and Advanced Applications - SEAA (Oulu, Finlandia, 2011). IEEE

[35] Martínez-Ruiz, T., García, F. and Piattini, M. Enhanced Variability Mechanisms to Manage Software Process Lines. Publizon, City, 2009.

[36] Martínez-Ruiz, T., García, F. and Piattini, M. Towards a SPEM v2.0 Extension to Define Process Lines Variability Mechanisms. Springer Verlag, City, 2008.

[37] Runeson, P., Höst, M., Rainer, A. and Regnell, B. Case Study Research in Software Engineering: Guidelines and Examples. John Wiley & Sons, 2012.

[38] Runeson, P. and Höst, M. Guidelines for Conducting and Reporting Case Study Research in Software Engineering. Empirical Software Engineering, 14, 2 2009), 131-164.

[39] Jacobson, I., Booch, G. and Rumbaugh, J. The Unified Software Development Process. Pearson Education, 1999.

[40] Schwaber, K. and Sutherland, J. Scrum Guide. City, 2010.

[41] Schwaber, K. and Beedle, M. Agile Software Development with Scrum. Addison-Wesley, 2001.

[42] Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. Agile Manifesto. City, 2001.

[43] Woodward, E., Ganis, M. and Surdek, S. A Practical Guide to Distributed Scrum. City, 2010.

[44] Larman, C. and Vodde, B. Practices for Scaling Lean & Agile Development. Pearson Education, 2010.

[45] IBM Eclipse Process Framework Composer. City, 2012.