

Towards a requirements reuse method using Product Line in distributed environments

Thais Ebling, Jorge Luis Nicolas Audy, Rafael Prikladnicki
Pontifical Catholic University of Rio Grande do Sul (PUCRS), Porto Alegre, Brazil
thais.ebling@pucrs.br, audy@pucrs.br, rafaelp@pucrs.br

Abstract

Distributed Software Development (DSD) is a recent approach where the teams are geographically distributed. Some characteristics of these environments have significant impact in activities that require constant communication, shared vision and stakeholder's cooperation, as we have in Requirements Engineering (RE).

The goal of this paper is to present a requirements reuse method that integrates software reuse in the context of Product Lines (PL), to improve the RE in a DSD environment.

1. Introduction

Requirements Engineering in Distributed Software Development presents several challenges. The reuse approach proposes a systematic set of processes, techniques and tools to reuse software artifacts. The reuse at the requirements level is among the promising ways to increase the reuse benefits [22], but there is no evidence in the literature about the possible benefits of reusing requirements as a strategy to improve RE in DSD. The Product Line approach supports the requirements reuse, through identification and reuse of features and requirements of the company's domain [12].

In this paper we propose a requirements reuse method using PL in DSD, that explores the integration of Requirements Reuse, Product Lines and Requirements Engineering aiming at the improvement of RE in DSD. We believe that our proposal can be particularly useful for DSD organizations that need a well defined domain as a basis for the development of many similar applications.

This study was partially supported by the Research Group on Distributed Software Development of the PDTI financed by Dell Computers of Brazil Ltd. (Law 8.248/91).

The contribution of this paper relies on presenting our proposal of a requirements reuse method using PL in DSD.

This paper is organized as following: in Section 2 we present the challenges of RE in DSD; in Section 3 we present the main concepts of the software reuse, requirements reuse and the PL approach; in Section 4 we present our proposal to requirements reuse using PL in DSD; in Section 5 we present the next steps.

2. RE in DSD environments

DSD is an approach where teams are geographically distributed [1]. Some characteristics of DSD (physical and temporal distance, cultural and language differences [4]) impact mainly in activities that require constant communication, shared vision and stakeholder's cooperation (e.g. Requirements Engineering).

A previous systematic review [23] has identified several challenges of RE in DSD, including:

Communication issues: Geographic dispersion introduces time differences and makes it hard the communication of teams [4],[24],[7] and the lack of informal communication negatively impacts relationship building [8].

Lack of common understanding of requirements: In DSD, the difficulties of achieving a common understanding about the requirements are amplified [7],[9],[25]. The lack of common understanding can lead to requirements misinterpretation [26],[8] and reduced collaboration between stakeholders [4].

Lack of collaboration: The challenges in collaboration between distributed teams happen due to differences in culture, language, organization processes [4],[8],[5].

Lack of common goals: In distributed environments it's hard to establish common goals, due to problems in communication, lack of common

understanding, cultural differences, etc. [5]. This can cause different viewpoints and priorities in the development process [24],[4].

National and organizational cultural differences: Cultural differences can create several challenges for achieving a shared understanding of the requirements [7] and often result in remote stakeholders' misinterpretation [8],[4], particularly in distributed requirements analysis [24],[27]. The distance aggravates the cultural gap between the requirements and development teams [4],[26]. Cultural differences are the reason of the use of multiple RE processes and tools which can be problematic in a distributed RE context [5],[7],[6],[8].

Change Management issues: Change Management can be a daunting task in RE of DSD environments [25],[5],[6],[9],[28],[8]. Changes may cause an extensive process of analysis and communication between distributed teams.

Knowledge Management issues: DSD makes it more difficult to seek out and to integrate knowledge about requirements [24]. The information is not appropriately shared with the distributed stakeholders [4] and the interaction between them is affected [8].

Lack of efficient tools and techniques: RE in DSD requires new or extended techniques to support distributed development tasks; and efficient distributed RE management [26]. It's difficult to track discussions on requirements that are stored across several media, due the lack of existing requirements tools that provide support for collaboration [25].

Many authors have researched about these challenges, but there is no evidence in the literature about the possible benefits of reusing requirements as a strategy to improve RE in DSD. Our proposal is to explore the possible benefits of integrating Requirements Reuse, PL and Requirements Engineering aiming at the improvement of RE in DSD. Next section presents the main concepts related to software reuse, requirements reuse and the PL approach.

3. Reuse and Product Lines

Software reuse is the use of existing software or software knowledge to construct new applications [29]. The reuse at the requirements level is among the promising approaches [22]. It can help the RE process in the following ways: reducing the time-to-specification; supporting the completeness checking of new requirements; sharing knowledge; providing reuse at later stages of development; helping in estimate cost and effort; and reducing uncertainty [30].

Product Line approach supports requirements reuse, through identification and reuse of features and requirements of the company's domain [12]. First, the common and variable requirements of the PL products are identified and analyzed, then their variabilities are documented with variation points which are filled to create product requirements [2].

Organizations that have succeeded with PL vary widely [2]. Nevertheless, there are universal activities:

Core Asset Development: Also called "Domain Engineering". Establishes a reusable platform and defines the PL commonality and variability [13].

Product Development: Also called "Application Engineering". The PL products are derived from the platform established.

Management: Technical and organizational management plays a critical role in the successful of a PL [2].

In the next section we present our method.

4. A requirements reuse method using PL in DSD environments

Our method was proposed based on an extensive literature review. To help with the identification of each research area that guided our proposal, we used the following labels:

- "*PL*" for the research area about RE in PL;
- "*DSD*" for the research area about RE in DSD;
- "*RW*" for related works (requirements reuse using PL in DSD);
- "*OP*" for our proposal. Presents the proposals of our method related to activities, artifacts and roles, based on the research areas.

As related works, we have the study of Gao et al [11] that presents experiences and challenges to develop PL tools that share information to distribute teams, but it's not specific to requirements reuse. Thurimella and Wolf [10] that proposes a variability model and justification matrices to help the requirements communication in DSD. And finally, the study of Cho [14] that proposes some notations to identify variations and dependencies of the PL requirements (PLR) and a scheme to reuse the PLR, but it's not focused in DSD.

Our proposal consider the contributions of the related works and include the focus on collaborative aspects of RE in DSD, establishing a path for requirements reuse in distributed environments. It consists of five disciplines and each discipline consists of activities that produce artifacts and are performed by roles.

4.1. Roles

Our method includes the proposal of roles to be responsible for the execution of each activity.

Customers

PL: Customers provide features and qualities for the products [15]. They have specific needs, which it's exploited to derive PL products [13].

DSD: Customers are individuals or companies that requested the project [17].

RW: The PL needs to support the instantiation of the products based on the concerns of the customers [10].

OP: The customers are individuals or companies, possibly geographically distributed, that provide the features and requirements of the products.

PL Manager

PL: The Manager must be a visionary leader that keeps the organization pointed toward PL goals [2]. The Executive role manages business goals [12].

OP: The PL Manager plans, manages and takes initial decisions of the distributed PL.

Product Manager

PL: This role involves the planning and evolution of the products [3]. He analyses the possibility and the actions to be taken for inclusion and exclusion of products [13].

OP: The Product Manager interacts with distributed teams to plan and manage the company's products.

Reuse Manager

PL: Technical management ensures that those who build core assets and products are engaged in the required activities [2]. System integrators are responsible for the quality and for acceptance criteria requirements [12].

OP: The Reuse Manager manages the reuse process and the use of the tool, to ensure that the artifacts management strategy is being followed.

Change Manager

PL: Domain asset manager maintain the versions and variants of the domain assets [3].

RW: Change Control Board team reviews and approves the PL changes via teleconference [14].

OP: The Change Manager controls the changes of the PL artifacts. He monitors the change process.

Domain Requirements Engineer

PL: Domain Requirements Engineer identifies domain requirements and their variability [13]. He take care of the evolution of the family [16].

DSD: Requirements Engineer is responsible for elicitation, analysis, negotiation, documentation, validation and management of requirements [17].

RW: Domain engineers help the instantiation of the variation points [10].

OP: The Domain Requirements Engineers, which may be co-localized or distributed, identifies, analyzes and documents the domain features and requirements.

Domain Collaborator/Product Collaborator

PL: Collaboration specialists are people with good communication skills that have knowledge about the domain and the products [3].

DSD: Ambassadors help the collaboration and communication about requirements [17]. Human facilitator helps the requirements decision-making meetings and manages conflict [4].

OP: The Domain and Product Collaborator help the communication between distributed teams, solving questions about domain and products.

Product Requirements Engineer

PL: Application Requirements Engineer develops and maintains the requirements for a single product [3]. He produces family members to satisfy customer's requirements [16].

DSD: Requirements Engineer is responsible for elicitation, analysis, negotiation, documentation, validation and management of requirements [17].

RW: Application engineers help the instantiation of variation points [10]. Regional marketing representatives are organized worldwide to promote new products and collect regional customer's requirements [14].

OP: The Product Requirements Engineers, possibly distributed, identifies the product features and requirements and reuses the domain artifacts.

4.2. Artifacts

Each activity produces artifacts, presented next:

Tool

PL: The PL tool must support the development process and organizational structure, enforce PL integrity, represent products capabilities, maintain traceability, support architecture-based development with multiple views, provide insight into business and technical decisions, incorporate new technologies [2].

DSD: The groupware tools must help the interaction and increase the understanding of the requirements documents [17].

RW: Using a repository, all involved teams on a global project can share information. To cope with diverse needs, it's essential to provide customizable

data formats and report templates. The tool must send notifications to users [11]. The product requirements are created with help from a front-end tool that guides this process [14].

OP: The tool for reuse requirements using PL in DSD must have (at least) these features: accessibility; interoperability; systematization of the reuse process; creation and search of artifacts; traceability; configuration management; management reports; documentation of reuse experiences; notifications to users; collaboration and interaction.

PL Plan

PL: The PL Adoption Plan describes the business goals and the strategy of the company [19].

OP: The PL Plan is created during the PL adoption and contains the organizational definitions. In our proposal, this artifact (that we assume that the organization already has) describes: PL scope, products and domains, company's goals, etc. During the activities proposed by our method, this Plan will help the understanding of the PL context.

Employees Register

DSD: To know as much as possible about the requirements elicitation scenario, it's important collect information about distributed teams [20].

OP: The Employees register include personal information (name, nickname, birthday, hobby, photo); professional information (local work description, email, phone, knowledge about process, technologies, products); and cultural information (native country and language, knowledge in others languages, cultural influences).

PL Role Plan

PL: The map of PL activities and roles determines the amount to which people work together [3].

DSD: RE in DSD requires knowledge about the roles and activities [15]. It requires a clear definition of these aspects [6].

OP: The PL Role Plan identifies the members of distributed teams and their roles. This will help the collaboration in distributed environments.

PL Dictionary

PL: The Dictionary defines the terminology utilized in the work products and supports a consistent view of the PL requirements [12].

DSD: A Dictionary can help to solve language questions [15] and to share a common vocabulary in DSD [20].

OP: The PL Dictionary identifies terminologies of products, domains and artifacts, and also define a common vocabulary to distributed teams.

Cultural Base

DSD: Cultural Base provides knowledge to distributed teams about the context where the software will be used [17]. This helps the RE in DSD [21].

OP: The Cultural Base helps the creation of the PL artifacts, storing cultural information about the context where the products will be used. This information include: writing and language issues, measures, cultural influences about products, context of use, etc.

Patterns Plan

DSD: RE in DSD requires the creation of requirements specification patterns [18] and is suggested the sharing of requirements-specification templates [5].

RW: It's essential to provide customizable data formats and report templates [11].

OP: The Patterns Plan identifies patterns to create the PL artifacts, the structure and content of these artifacts. This will provide a common documentation of distributed teams.

Artifacts Management Plan

PL: System integrators are responsible for quality in requirements and for acceptance criteria [12].

OP: The Artifacts management Plan identifies the criteria for artifacts management. Include criteria for artifacts acceptance, evaluation, classification, exclusion, inclusion, etc.

Experiences Register

PL: One PL practice is the documentation of existing and well-proven PL experiences [19].

DSD: The information generated through the RE process must be shared to distributed teams [18].

RW: All involved distributed teams must share information of global software projects [11].

OP: The Experiences register describes the reuse experiences from distributed teams, including learned lessons, good and bad practices, etc.

Management Reports

PL: Technical management collects data to track progress [2].

OP: The Management reports present strategic information to the organization (verification of reuse goals, identification of market trends and opportunities, etc).

Domain Artifacts

PL: Domain Requirements Engineering is responsible for development of common and variable requirements and their precise documentation [3].

RW: The specification of the PL requirements should describe the requirements of the core asset and their variations [14]. To help the understanding of variability in domain artifacts are suggested the identification of rationales [10].

OP: The domain artifacts include models that document the common features and requirements of the PL and their variability. To help the reuse process and the understanding of the PL context can be used the documentation of rationales. Domain artifacts can include: Feature Model, Use case Model, Orthogonal variability Model, Requirements Specification, etc.

Product Artifacts

PL: Application artifacts comprise all development artifacts of a specific application [13].

RW: Program Manager creates a project file of PR for a new product which specifies requirements of base model and its series [14].

OP: The product artifacts include models that document the specific features and requirements of a product. Usually they are instances of domain artifacts.

4.3. Disciplines

Our proposal establishes a path for requirements reuse in DSD. The first iteration starts with the execution of the "Initial definitions" discipline. In order to reuse the domain requirements on creation of products, the "Definition of domain requirements" discipline must be executed before the "Definition of product requirements" discipline. In the next iterations, the company can start and run the disciplines in any order. The Figure 1 presents our method:

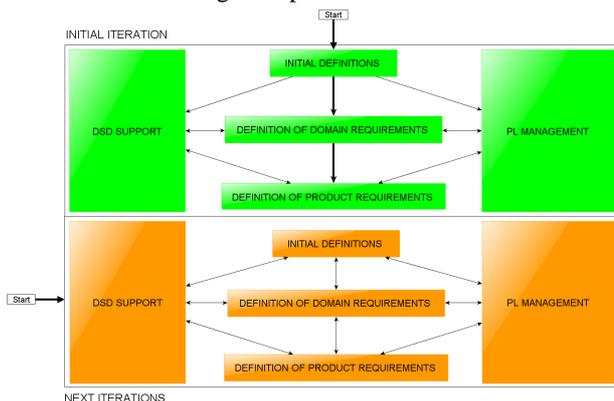


Figure 1. Our method

Our approach defines a path to follow, but still supports individual choices of the organization. Next we present the activities for each discipline of the proposed method.

4.3.1. Initial Definitions

This discipline establishes initial definitions to reuse requirements using PL in DSD. The Figure 2 presents this discipline:

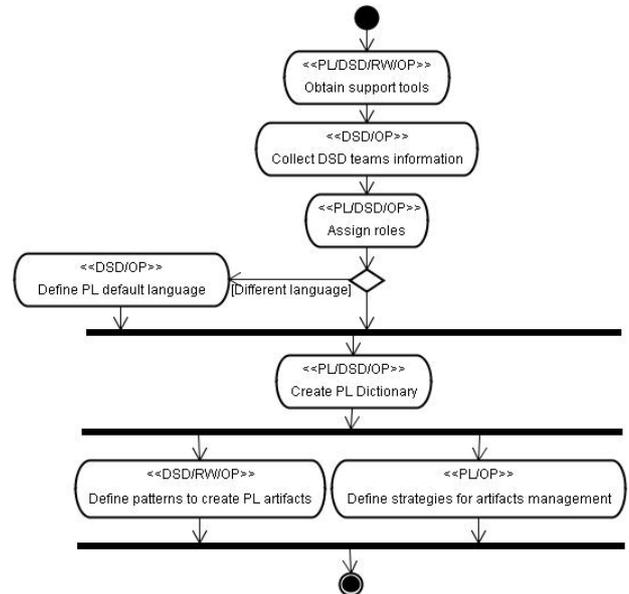


Figure 2. Initial definitions

Next we present the activities of this phase:

Obtain support tools

PL: The infrastructure for turning out a software product requires specific PL processes and appropriate tool support [2].

DSD: The groupware tools must help the interaction and increase the understanding of the requirements documents [17].

RW: Using a repository the teams of global projects can share information. To cope with diverse needs, it's essential to provide customizable data formats and report templates. In global environments the tool must send notifications to users [11]. The product requirements are created with help from a front-end tool [14].

OP: It's necessary to obtain support tools to help the artifacts development and the reuse process, share the artifacts and experiences to all stakeholders and help the interaction and collaboration of distributed teams.

Collect DSD teams information

DSD: To know about the requirements elicitation scenario, it's important collect information about distributed teams [20].

OP: To help the interaction and understanding between distributed stakeholders and also the allocation of roles, we suggest the collection of

personal, professional and cultural information of teams.

Assign roles

PL: Often we see a single domain group and several separate application groups [3].

DSD: RE in DSD requires knowledge about the roles and activities [17].

OP: In this activity are assigned the roles to distributed teams. The professional information collected in the previous activity can be used to define professional profiles.

Define the PL default language

DSD: To RE in DSD must be defined the language that will be used to create the artifacts [17].

OP: To reduce the communication problems and to increase the understanding about the artifacts, we suggest the definition of a default language.

Create the PL Dictionary

PL: The Dictionary defines the terminology utilized in the work products and supports a consistent view of the PL requirements [12].

DSD: A Dictionary can help to solve language questions [17] and to share a common vocabulary [20].

OP: The PL Dictionary helps to solve language questions and provides an overview of the PL artifacts.

Define patterns to create PL artifacts

DSD: RE in DSD requires the creation of requirements specification patterns [17]. It's suggested sharing of requirements-specification templates [5].

RW: It's essential to provide customizable data formats and report templates [11].

OP: In this activity, we suggest the definition of patterns to create the PL artifacts.

Define strategies for artifacts management

PL: System integrators are responsible for quality in requirements and for acceptance criteria [12].

OP: In this activity are defined the strategies for artifacts management.

4.3.2. Definition of domain requirements

In this discipline the organization's core asset are created. The Figure 3 presents this discipline:

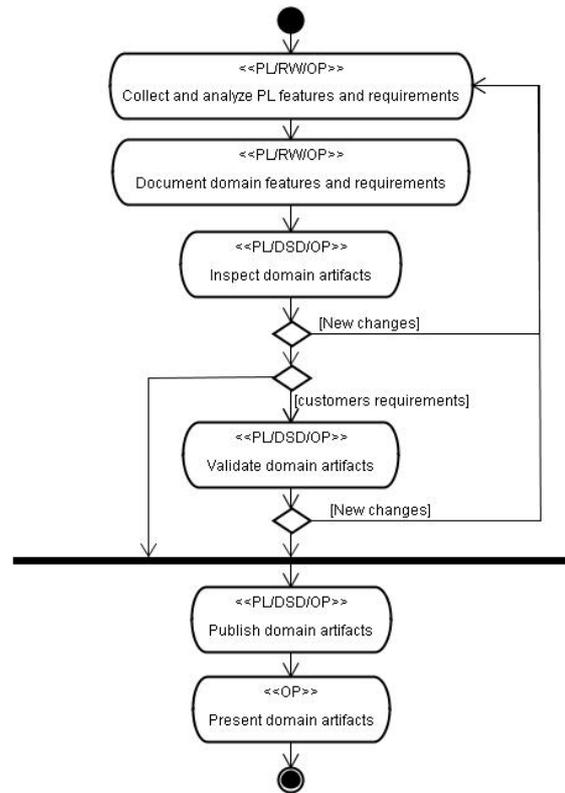


Figure 3. Definition of domain requirements

Next we present the activities of this phase:

Collect and analyze PL features and requirements

PL: The Domain Requirements Engineering encompasses elicitation and documentation of common and variable requirements [13]. It comprises the identification of common and variable aspects among family members [16].

RW: The PL variability identification may involve the collaboration of stakeholders from application engineering [10].

OP: This discipline includes the collection of the PL features and requirements through the existing products and documentations or through customer's elicitation in distributed sites. Then, the PL requirements and features are analyzed to identify their variability.

Document domain features and requirements

PL: Common requirements are written with variation points that can be filled to create product-specific requirements [2]. The Domain Requirements Engineering develops the common and variable requirements and their precise documentation [3].

RW: The specification of PL requirements should describe requirements of the core asset and their variations [14]. To help the understanding of the variability in domain artifacts are suggested the identification of rationales [10].

OP: In this activity, the features and requirements are documented, identifying their variability. The traceability must be maintained. To help the reuse process and the understanding of the PL context can be used the documentation of rationales.

Inspect domain artifacts

PL: The common PL requirements must be verified [2] to ensure their accuracy and completeness [12].

DSD: The requirements must be inspected to ensure their understanding to all distributed teams [17] and must be analyzed in order to determine consistency between the different statements [20].

OP: The domain artifacts must be inspected to ensure their consistency, quality and understanding in distributed environments. They can be changed if necessary.

Validate domain artifacts

PL: The PL requirements must be verified [2].

DSD: The requirements must be validated to ensure that they meet the customer’s goals [17].

OP: The domain artifacts must be validated, ensuring that they meet the customer’s needs. They can be changed if necessary.

Publish domain artifacts

PL: Domain artifacts are stored in a common repository [13].

DSD: RE in DSD requires a repository of artifacts [4]. It’s necessary share a project-artifacts [5].

OP: After the inspection and validation, the domain artifacts can be published to be reused.

Present domain artifacts

OP: To disseminate knowledge about the domain artifacts they must be presented to the distributed teams.

4.3.3. Definition of product requirements

In this discipline the products of the organization are created through the reuse of the core assets. The Figure 4 presents this discipline:

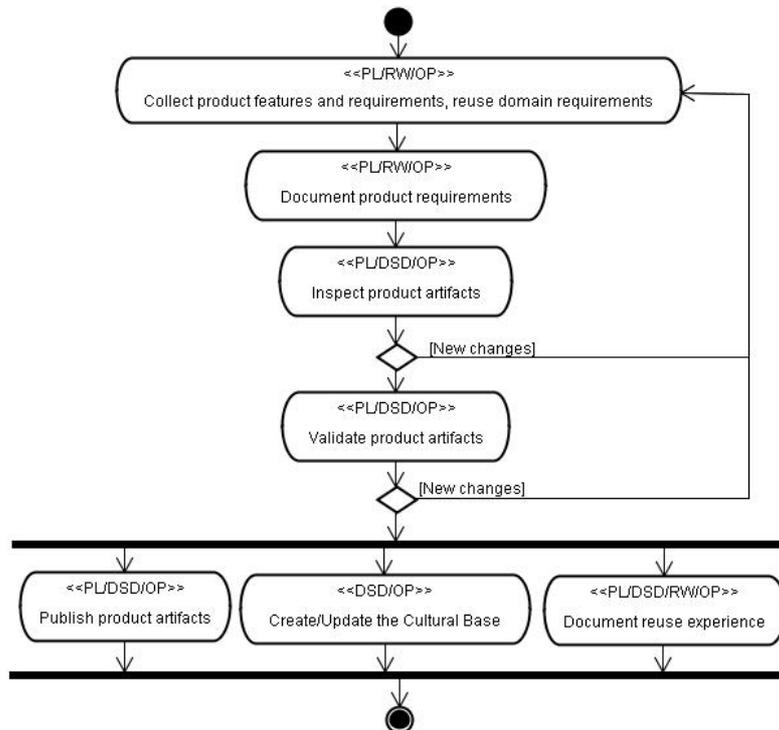


Figure 4. Definition of product requirements

Next we present the activities of this phase:

Collect product features and requirements, reuse domain requirements

PL: The Application Engineering must achieve an as high as possible reuse of the domain assets; exploit the commonality and variability of the PL during the development of a application; document the application artifacts; bind the variability according to the application needs; estimate the impacts of the differences between application and domain requirements artifacts [13].

RW: The PL variability identification may involve the collaboration of stakeholders from application engineering [10]. Regional marketing representatives are organized worldwide to promote new products and to collect regional customer's requirements [14].

OP: In this activity, first the product features and requirements are obtained through changes on domain artifacts or customer's elicitation. Then, the domain artifacts are verified, to analyze if exists a domain requirement that satisfy the customer's needs. There are three outputs for this analysis: (i) Direct reuse - there are domain artifacts which fully meet the customer needs and can be reused; (ii) Indirect reuse - there are domain artifacts which partially meet the customer needs; (iii) No reuse – there aren't domain requirements that satisfy the customer needs.

Document product requirements

PL: The Application Requirements Engineering encompasses all activities for developing the application requirements specification [13].

RW: Program Manager creates a project file of product requirements for a new product which specifies requirements of base model and its series [14].

OP: The product requirements are documented and the traceability must be maintained.

Inspect product artifacts

PL: The product-specific requirements must be verified [2], ensuring their accuracy and completeness [12].

DSD: The requirements must be inspected to ensure their understanding to distributed teams [17] and to determine consistency between different statements [20].

OP: The product artifacts must be inspected to ensure their consistency, quality and understanding in distributed environments. They can be changed if necessary.

Validate product artifacts

PL: The product-specific requirements must be verified [2].

DSD: The requirements must be validated to ensure that they meet the needs and goals of the customer [17].

OP: The product artifacts must be validated, ensuring that they meet the customer's needs. They can be changed if necessary.

Publish product artifacts

PL: PL artifacts are stored in a repository [13].

DSD: RE in DSD requires a repository of artifacts [4]. It's necessary share a project-artifacts repository [5].

OP: After the inspection and validation, the product artifacts can be published.

Create/Update the Cultural Base

DSD: Cultural Base provides knowledge to distributed teams about the context where the software will be used [17], helping the RE in DSD [21].

OP: In DSD environments, the cultural differences are the reason of several difficulties and misunderstandings, especially during the RE. To reduce these problems, the organization can create a Cultural Base.

Document reuse experience

PL: One PL practice is the documentation of existing and well-proven PL experiences [19].

DSD: The information about requirements process must be shared to distributed teams [18].

RW: All involved distributed teams must share information of global software projects [11].

OP: In this activity the reuse experiences are documented. To distributed environments, this will encourage the reuse and share knowledge.

4.3.4. DSD Support

This discipline supports the distributed software development. The Figure 5 presents this discipline:

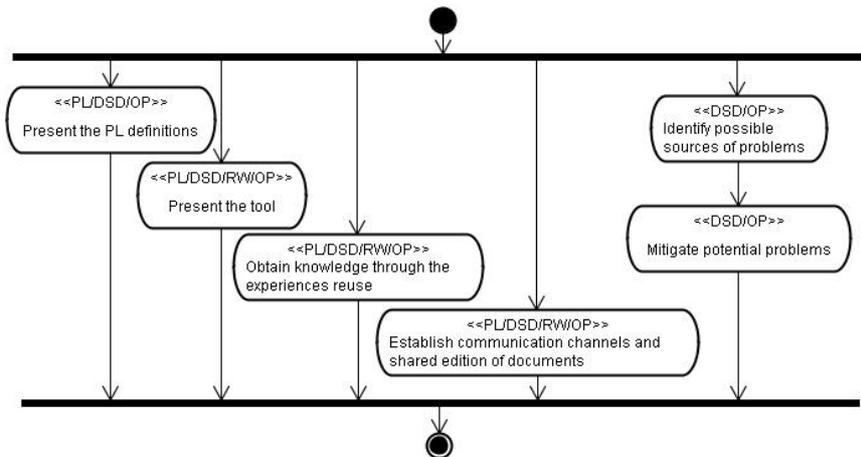


Figure 5. DSD Support

Next we present the activities of this phase:

Present the PL definitions

PL: Teams must be trained beyond general software engineering and corporate procedures to ensure that they understand PL practices [2].

DSD: RE in DSD requires training about the processes, tools and technologies [5]. It's suggested face-to-face relationship and training prior to project initiation [7].

OP: We suggest the execution of training, courses or workshops to present the PL definitions to distributed teams.

Present the tool

PL: Establishing tool support for a PL includes training tool users and maintainers [2].

DSD: RE in DSD requires training about the processes, tools, and technologies [5]. It's suggested face-to-face relationship and training prior to project initiation [6].

RW: To PL in distributed environments, is suggested the face-to-face training to users and customers [11].

OP: We suggest the execution of training, courses or workshops to present the PL tool to distributed teams.

Obtain knowledge through the experiences reuse

PL: One PL practice is the documentation of existing and well-proven software PL experience [19].

DSD: The information generated through the RE process must be shared to distributed teams [18].

RW: All involved distributed teams must share information of global software projects [11].

OP: The documentation of reuse experiences helps to increase the knowledge sharing, especially in DSD

teams. In anytime, anyone of distributed teams can obtain knowledge through this documentation.

Establish communication channels and shared edition of documents

PL: The PL in distributed environments requires extensive communication, both through face-to-face and virtual meetings such as telephone conferences and video meetings [3].

DSD: RE in DSD requires use of electronic mediation [5]. To improve communication in RE of DSD, is suggested an analysis about how the technologies selection influence people performance [20].

RW: In distributed PL, the Change Control Board team reviews and approves the changes via teleconference [14].

OP: To help the collaboration and communication of distributed teams, the teams can establish communication channels and shared edition of documents. Communication channels can be used in several activities, like: negotiation and elicitation of requirements, artifacts inspection, etc. Shared edition of documents can be used to create and update PL documents and models.

Identify possible sources of problems

DSD: One suggested practice to RE in DSD is the identification of possible problems and the recommendation of strategies to improve the requirements process [20].

OP: We suggest the identification of possible sources of problems, like: cultural and temporal differences between distributed teams; lack of knowledge about domains, products, requirements, process, technologies; etc.

Mitigate potential problems

DSD: One suggested practice to RE in DSD is the identification of possible problems and the recommendation of strategies to improve the requirements process [20].

OP: To reduce the problems identified in the previous activity, we suggest the execution of trainings, meetings and workshops.

4.3.5. PL Management

This discipline is responsible for the PL management. The Figure 6 presents this discipline:

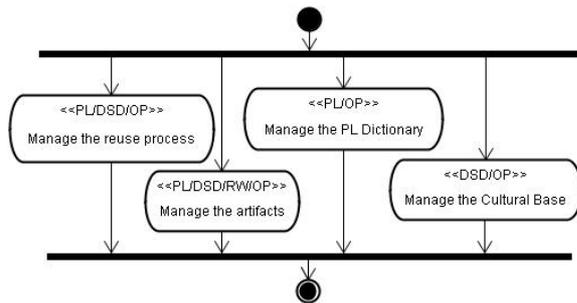


Figure 6. PL Management

Next we present the activities of this phase:

Manage the reuse process

PL: Management must direct, track and enforce the use of assets [2].

DSD: The RE in DSD requires practical performance metrics and project-reporting mechanism [5].

OP: The reuse process has to be managed to ensure that those who build core assets and products are engaged in the required activities. Through the management of the PL, the organization can obtain strategic information.

Manage the artifacts

PL: Change management policies must provide a mechanism for proposing changes and supporting the systematic assessment of how these changes will impact the PL [2].

DSD: The RE in DSD requires the management of requirements changes and the analysis the changes impact [17].

RW: In distributed PL, the Change Control Board team reviews and approves the changes [14].

OP: The PL artifacts also need management. The change management in PL and distributed environments is essential and challenging. To change the artifacts, first is necessary to analyze the impact, interacting with distributed teams. The execution of changes in artifacts must follow the activities proposed by Discipline DEFINITION OF DOMAIN REQUIREMENTS (to change domain artifacts) or Discipline DEFINITION OF PRODUCT REQUIREMENTS (to change product artifacts).

Manage the PL Dictionary

PL: The Dictionary persists for the PL lifetime [12].

OP: The PL Dictionary has to be constantly managed.

Manage the Cultural Base

DSD: The Cultural Base has to be managed [17].

OP: The Cultural Base has to be constantly managed.

4.4. Overview of our proposal

Table 1 presents an overview of our method, including the activities of each discipline, their input and output artifacts and also the roles involved:

Table 1. Proposal overview

ACTIVITY	INPUT	OUTPUT	ROLES
DISCIPLINE INITIAL DEFINITIONS			
Obtain support tools	No	Tool	PL Manager
Collect DSD teams information	Tool	Employees register	PL Manager; DSD teams
Assign roles	Tool; Employees register	PL role Plan	PL Manager; DSD teams
Define the PL default language	Tool; Employees register	PL Plan updated	PL Manager
Create the PL Dictionary	Tool; PL Plan	PL Dictionary	PL Manager; Domain and Product Collaborator
Define patterns to create the PL artifacts	Tool; PL Dictionary	Patterns Plan	PL Manager; Domain and Product Collaborator
Define strategies for artifacts management	Tool; PL Dictionary	Artifacts management Plan	PL Manager; Domain and Product Collaborator
DISCIPLINE DEFINITION OF DOMAIN REQUIREMENTS			
Collect and analyze PL features and requirements	Tool; PL Plan	No	Domain Requirements Engineer; Domain Collaborator; Customer; Product Requirements Engineer
Document domain features and requirements	Tool; PL Dictionary; Patterns Plan	Domain artifacts	Domain Requirements Engineer
Inspect domain artifacts	PL Dictionary; Patterns Plan; Artifacts management Plan; PL Plan; Domain artifacts	Domain artifacts updated	Domain Requirements Engineer; Domain Collaborator; Reuse Manager; Product Manager
Validate domain artifacts	Domain artifacts	Domain artifacts updated	Domain Requirements Engineer; Domain Collaborator; Customer

Publish domain artifacts	Tool; Domain artifacts; Artifacts management Plan	No	Domain Requirements Engineer
Present domain artifacts	Domain artifacts	No	Domain Collaborator; DSD teams
DISCIPLINE DEFINITION OF PRODUCT REQUIREMENTS			
Collect product features and requirements, reuse domain requirements	PL Plan; PL Dictionary; Domain artifacts	No	Product Requirements Engineer; Customer; Domain and Product Collaborator
Document product requirements	Tool; PL Dictionary; Cultural Base; Patterns Plan; Domain artifacts	Product artifacts	Product Requirements Engineer
Inspect product artifacts	PL Dictionary; Patterns Plan; Artifacts management Plan; PL Plan; Product artifacts	Product artifacts updated	Product and Domain Requirements Engineer; Product Collaborator; Reuse Manager
Validate product artifacts	Product artifacts	Product artifacts updated	Product Requirements Engineer; Product Collaborator; Customer
Publish product artifacts	Tool; Product artifacts; Artifacts management Plan	No	Product Requirements Engineer
Create/Update the Cultural Base	Tool	Cultural Base	PL Manager; Domain and Product Requirements Engineer
Document reuse experience	Tool; PL Dictionary;	Experiences register	Product Requirements Engineer
DISCIPLINE DSD SUPPORT			
Present the PL definitions	PL Plan; PL role Plan; PL Dictionary; Cultural Base; Patterns Plan; Artifacts management Plan	No	PL Manager; DSD teams
Present the tool	Tool	No	Reuse Manager; DSD teams
Obtain knowledge through the experiences reuse	Experiences register	No	DSD teams
Establish communication channels and shared edition of documents	Tool	No	DSD teams
Identify possible sources of problems	Employees register	No	DSD teams
Mitigate potential problems	Tool; PL Plan; PL Dictionary; Cultural Base	No	Domain and Product Collaborator
DISCIPLINE PL MANAGEMENT			
Manage the reuse process	Tool	Management reports	Reuse Manager; PL Manager; Product Manager
Manage the artifacts	Artifact that will be changed	Artifact changed	Change Manager; Product Manager; Domain and Product Requirements Engineer
Manage the PL Dictionary	PL Dictionary	PL Dictionary updated	PL Manager; DSD teams
Manage the Cultural Base	Cultural Base	Cultural Base updated	PL Manager; Domain and Product Requirements Engineer

4.5. Our proposal and the RE in DSD

In this subsection we present the main challenges of RE in DSD (as presented in Section 2) and how our method aims to reduce such challenges for companies where the PL approach is suitable:

- **Communication issues:** use of communication channels; definition of a default language; use of a PL Dictionary;
- **Lack of common understanding of requirements:** use of a PL Dictionary; use of patterns to create the PL artifacts; definition of strategies for artifacts management;
- **Lack of collaboration:** collection of personal, professional and cultural information of the teams; use of shared edition of documents; identification of those responsible for the tasks;
- **Lack of common goals:** presentation of PL definitions; management of the reuse process;
- **National and organizational cultural differences:** collection of cultural information of the teams; use of patterns to create the PL artifacts; definition of strategies for artifacts management; use of a Cultural Base;
- **Change Management issues:** change management; use of a tool to support it;

- **Knowledge Management issues:** obtaining knowledge through documentation of reuse experiences; use of a tool to share it; reuse of artifacts; presentation of domain artifacts;
- **Lack of efficient tools and techniques:** use of a tool focused in reuse and DSD environments;

Our proposal also includes goals related to software reuse, including: large-scale productivity gains, decreased time to market, increased product quality, etc.

5. Next steps

In this paper we have presented our proposal to reduce the existing challenges of distributed RE by integrating requirements reuse and Product Lines. Our method aiming at an improvement in the execution of distributed projects where a domain definition may be necessary for the development of applications.

As future works, we will (i) build a reuse policy, based on a literature review, which will suggest some techniques, methods, communication media and other aspects specific to distributed environments for each one of the activities, and (ii) evaluate the practical benefits of our proposal.

6. References

- [1] Carmel, E. "Global Software Teams – Collaborating Across Borders and Time-Zones". *Upper Saddle River, NJ, Prentice Hall*. 1999.
- [2] SEI - Software Engineering Institute. "A Framework for Software Product Line Practice, version 5". <http://www.sei.cmu.edu/productlines/framework.html>. 26-01-2009.
- [3] Linden, F. J. van der, Schmid, K., Rommes, E. "Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering". *Springer*. 2007.
- [4] Damian, D. E. and Zowghi, D. "The impact of stakeholders' geographical distribution on managing requirements in a multi-site organization". *Proc. of the 10th Anniversary IEEE Joint Int'l Conference on Requirements Engineering*. 2002.
- [5] Bhat, J. M., Gupta, M. and Murthy, S. N. "Overcoming Requirements Engineering Challenges: Lessons from Offshore Outsourcing". *IEEE Software*, volume 23, issue 5, page(s):38 – 44. 2006.
- [6] Berenbach, B. "Impact of organizational Structure on Distributed Requirements Engineering Processes: Lessons Learned". *Int'l Conference on Software Engineering*. 2006.
- [7] Herbsleb, J. D. "Global Software Engineering: The Future of Socio-technical Coordination". *Int'l Conference on Software Engineering*. 2007.
- [8] Damian, D. "Stakeholders in Global Requirements Engineering: Lessons Learned from Practice". *IEEE Software*, volume 24, issue 2, page(s):21 – 27. 2007.
- [9] Kommeren R. and Parviainen, P. "Philips experiences in global distributed software development". *Empirical Software Engineering*, volume 12, number 6. 2007.
- [10] Thurimella, A. K. and Wolf, T. "Issue-based Variability Modeling". *International Conference on Global Software Engineering*. 2007.
- [11] Gao, J. Z., Itaru, F. and Toyoshima, Y. "Managing Problems for Global Software Production – Experience and Lessons". *Information Technology and Management*, volume 3, issue 1-2, pages: 85 – 112. 2002.
- [12] Chastek, G., Donohoe, P., Kang, K. C. and Thiel, S. "Product Line Analysis: A Practical Introduction". *Technical Report CMU/SEI-2001-TR-001, ESC-TR-2001-001*. 2001.
- [13] Pohl, K., Böckle, G., Linden, F. J. van der. "Software Product Line Engineering: Foundations, Principles and Technique". *Springer*. 1998.
- [14] Cho, H. "Requirement Management in Software Product Line". *International Conference on Global Software Engineering*. 2007.
- [15] Chastek, G., Donohoe, P. "Product Line Analysis for Practitioners". *Technical Report CMU/SEI-2003-TR-008, ESC-TR-2003-008*. 2003.
- [16] Harsu, M. "FAST product-line architecture process". <http://practise2.cs.tut.fi/pub/papers/fast.pdf>. 26-01-2009.
- [17] Lopes, L. T. "Um Modelo de Processo de Engenharia de Requisitos para Ambientes de Desenvolvimento Distribuído de Software". *Dissertação de Mestrado. Pontifícia Católica Universidade do Rio Grande do Sul*. 2004.
- [18] Audy, J. L. N. and Prikladnicki, R. "Desenvolvimento Distribuído de Software: Desenvolvimento de Software com Equipes Distribuídas". *Editora Campus-Elsevier*. 2007.
- [19] Northrop, L. M. Software Product Line – Adoption Roadmap. *Technical Report CMU/SEI-2004-TR-022 ESC-TR-2004-022*. 2004.
- [20] Aranda, G. N., Vizcaíno, A., Cechich, A., Piattini, M. "A Methodology for reducing geographical dispersion problems during Global Requirements elicitation". *11th Workshop on requirements Engineering*. 2008.
- [21] Mahemoff, Michael J. and Johnston, Lorraine. Software Internationalisation: Implications for Requirements Engineering. *Proc. of the Third Australian Workshop on Requirements Engineering*. 1998.
- [22] Lam, W. "A case-study of requirements reuse through product families". *Annuals of Software Engineering*, volume 5, pages: 253 – 277. 1998.
- [23] Ebling, T, Audy, J. L. N and Prikladnicki, R. "A systematic literature review of requirements engineering in distributed software development environments". *Accepted for publication on 11th International Conference on Enterprise Information Systems*. 2009.
- [24] Berenbach, B. and Gall, M. "Toward a Unified Model for Requirements Engineering". *Int'l Conf on Global Software Engineering*, pages: 237 – 238. 2006
- [25] Sengupta, B., Sinha, V. and Chandra, S. "A Research Agenda for Distributed Software Development". *Int'l Conference on Software Engineering*. 2006.
- [26] Cheng, B. H.C. and Atlee, J. M. "Research Directions in Requirements Engineering". *Int'l Conference on Software Engineering*, pages 285-303. 2007.
- [27] Audy, J., Evaristo, R. and Watson-Manheim, M. B. "Distributed Analysis: The Last Frontier?". *Proc. of the 37th Hawaii Int'l Conference on System Sciences*. 2004.
- [28] Jacobs, J., Moll, J.V., Krausec, P., Kusters, R., Trienekens, J. and Brombacher, A. "Exploring defect causes in products developed by virtual teams". *Information and Software Technology*, volume 47, issue 6, pages 399-410. 2005.
- [29] Frakes, W. B. and Kang, K. "Software Reuse Research: Status and Future". *IEEE Transactions on Software Engineering*, volume 31, issue 7. Pages: 529 – 536. 2005.
- [30] Lam, W., Jones, S., Britton, C. "Technology Transfer for Reuse: A Management Model and Process Improvement Framework". *Proc. of the 3rd International Conference on Requirements Engineering: Putting Requirements Engineering to Practice*. 1998.