Towards a Global Software Development Community Web: Identifying Patterns and Scenarios

Miguel J. Monasor
Lero, The Irish Software Engineering Research Centre, University of Limerick
MiguelJ.Monasor@gmail.com

Aurora Vizcaíno
Alarcos Research Group, University of Castilla-La Mancha
Aurora.Vizcaino@uclm.es

Mario Piattini
Alarcos Research Group, University of Castilla-La Mancha
Mario.Piattini@uclm.es

John Noll
Lero, The Irish Software Engineering Research Centre, University of Limerick
John.Noll@lero.ie

Sarah Beecham
Lero, The Irish Software Engineering Research Centre, University of Limerick
Sarah.Beecham@lero.ie

***Abstract*— Sources of Global Software Development (GSD) information, such as academic literature, often focus on high-level issues rather than on specific problems. Researchers tend to generalize problems and solutions; however, practitioners and instructors frequently need to identify real low-level scenarios and patterns in an effort to study specific problems and their solutions.**

We propose a method for collecting and defining GSD scenarios and related patterns. Scenarios depicting events that happen in certain GSD contexts associated with communication, coordination are central to this method. In this paper we show how problems and solutions extracted from these events can lead to the definition of patterns. Patterns describe generalized information that can be re-used in similar contexts.

To facilitate knowledge sharing, we have integrated this pattern model into a GSD Community Web intended to promote collaboration between industry and academia. News, resources and discussion forums on GSD topics are also available through this website.

Keywords- global software development; distributed software development; community; patterns; scenarios

I. INTRODUCTION

Global Software Development (GSD) is a broad field in which different kinds of problems related to communication, coordination or software development processes emerge. Moreover, solutions to these problems depend very much on specific context and project settings [1]. Practitioners, instructors and researchers in the GSD field frequently need to identify real low-level problems, as well as to study specific scenarios and their solutions. Access to this information is not always easy, however, as the main public source of information is the published academic literature. Although the academic literature does include reports of specific case studies which are context specific, in many cases the specific context is lost as researchers focus on high-level issues in an effort to generalize problems and solutions.

In a previous study, we aimed to provide GSD training by simulating realistic distributed settings [2]. However, during this process we found that simulating accurate GSD scenarios requires access to real problems that could be reproduced. Apart from that, the kind of scenarios that we were looking for, based on cultural, linguistic and procedural problems in GSD, are perhaps too specific to be reported in the literature. This meant that we were confronted with the problem of not being able to gain access to the type of low-level, realistic problems and scenarios needed for a simulation-based training platform.

Moreover, this problem is not restricted to the educational field. Researchers could also benefit from having access to real GSD scenarios, along with information that come from previous experiences. Companies could also identify and classify common problems, so as to apply accepted solutions to improve their processes and conduct mitigating actions [3].

In this work we propose a method for defining GSD scenarios, and the patterns arising in them, with respect to problems and solutions related to communication, coordination or software development processes in the GSD field. *Scenarios* represent real events that happen in a certain GSD context. The problems and solutions that are extracted from these events lead to the definition of patterns. *Patterns* are the result of processing and generalizing the information that has come into being in a way that can be reused in similar contexts. Scenarios could therefore be associated with existing patterns.

In 2004, Hargreaves et al. [4] reported the need to build a strong GSD research community, and envisioned a website for the exchange of documentation, articles, project information, events, research techniques, models and theories. No further work on this has been reported to date, however. To tackle this issue, a GSD Community Web is also presented in this paper, the aim of which is to provide support to the pattern-based model. By means of this website, members of the community can contribute their knowledge, and provide patterns and scenarios.

II. RELATED WORK

When identifying problems and solutions it is helpful to use patterns [5]. *Patterns* describe problems that usually occur in an environment. They present the core of the solution to that problem in such a way that this solution can be used in the future to tackle problems that may follow the same pattern in a similar context [6]. One way to create a shared understanding of the problems that can happen in the

978-0-7695-5055-8/13 $26.00 © 2013 Crown Copyright
DOI 10.1109/ICGSEW.2013.13

different contexts of GSD is through extracting common patterns that include strategies on how to tackle these problems. For example, the Design Patterns contribution from the Gang of Four [7] became very important in the field of Software Design education. Another relevant work was conducted by Coplien and Harrison [8] who present the concept of Organizational Patterns, considering the structure of organizations and providing solutions to frequent software development problems.

In this work we focus on the definition of "**Pedagogical Patterns**" which aim to "capture the essence of the practice in a compact form that can be easily communicated to those who need the knowledge" [9]. These patterns can be used to apply common practices and techniques that regulate the flow of the collaborative learning activities. Based on these patterns, teachers can build up their own range of learning modules from a repository of collected experiences that can be reused [10]. **Antipatterns** are also applied, to define practices that produce more negative consequences than beneficial results [11].

A. GSD Patterns in Industry and Research

Table 1. Patterns in GSD

Cultural patterns
Yes (but no) pattern, proxy pattern, we'll-take-you-literally (anti) pattern, we're-one-single-team (anti) pattern, the-customer-is-king (anti) pattern [12].
Unproductive productivity, hesitant to always say yes, owning rather than modularizing [13].
Communication and interaction patterns
Four main types are identified by Paasivaara et al. [14]: 1) problem solving, 2) informing, monitoring and feedback, 3) relationship building, and 4) decision making and coordination.
Patterns for project management
GSD strategy, fuzzy front end, communicate early, divide and conquer with iterations, key roles in sites, communication tools, common repositories and tools, work allocation, architectural work allocation, phase-based work allocation, feature-based work allocation, use common processes, iteration planning, multi-level daily meetings, iteration review, organize knowledge transfer, manage competence and notice cultural differences [15].
Testing patterns
Test cases as memorandum of understanding of knowledge transfer, light-weight pre-acceptance test, communication on eye level, use tool for bug tracking, moving on-/off-business analyst, complementing testing attitudes, align understanding of general testing approach, continuous integration test, mirrored team manager, extension of the day, traceable test cases, tester's sparring partner, complement test skills, synchronized test environments, central test environment, evaluation of constraints for test data [16].
Requirements engineering patterns
Define 'use case', onboard business analyst during requirements engineering, ship test cases, use work packages and handover checkpoints, use bidirectional cross references and map business terms to entity attributes [17].

The literature deals with different kinds of patterns and models in several fields of application related to GSD, some of which are summarized in Table 1.

III. MODEL FOR DEFINING PATTERNS AND SCENARIOS

In this work we propose a model that aims to identify pedagogical patterns from relevant GSD scenarios in order to promote their reuse. To give support to this model, we provide a repository that can be used in software process improvement, research and teaching, as well as for training purposes in GSD. GSD patterns will be gathered by reviewing the literature and GSD community (researchers, practitioners and instructors) participation. This knowledge will therefore improve iteratively and be contrasted with expert opinions and experiences. The model aims to fulfill the following objectives:

1. Allow the GSD community to incorporate iteratively new scenarios and patterns that could be validated and reused.
2. Facilitate the sharing and reusing of patterns and scenarios.
3. Provide a platform for the promoting of the collaboration of the GSD community.

A. Defining GSD patterns

GSD involves a wide variety of problems in different fields; consequently, different kinds of GSD patterns can be derived. The collaboration of a community in the gathering of this variety of knowledge makes it necessary to define a common and simple way of representing and storing accurate patterns.

To define patterns, we follow a hierarchical model based on abstraction, in which the lower levels of the hierarchy make use of their own attributes, along with the attributes defined in the upper levels. On the basis of the patterns found in our literature review, Figure 1 sets out a summary, which includes some of the patterns defined at a first stage following this hierarchical model. The first level defines the common attributes that all the patterns will share:

- **Pattern name**: descriptive title of the pattern.
- **Problem**: detailing the problem as well as the conditions in which it can happen.
- **Population**: population that could be affected.
- **Analysis of the problem**: detailing information about the context or consequences of the problem.
- **Solution**: including practices that can avoid or mitigate the problem.
- **Reference/Source**: patterns can be gathered from literature or other sources.

These attributes are similar to those employed in by Coplien and Harrison [8] who apply: name, context, statement of the problem and solution. However, as context is important in GSD we have divided context into the following attributes: population and analysis of the problem, with the aim of providing details about the population and consequences in that context.

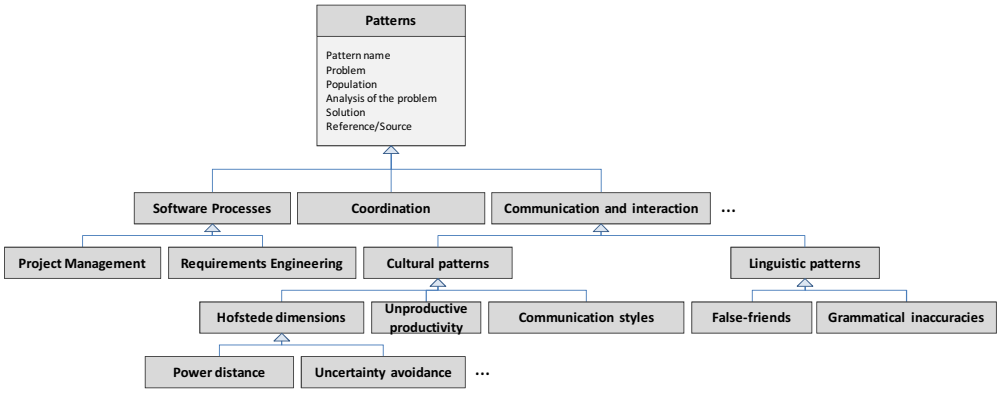

Figure 1. Pattern definition model

These attributes should enable any kind of GSD pattern to be defined. For example, **cultural patterns** representing common problems in the communication field include:

- Hofstede cultural dimensions [18]: power distance, individualism, uncertainty avoidance, masculinity and long term orientation.
- Communication styles: direct/indirect, formal/informal.
- Unproductive productivity [13].

Table 2 shows how the *unproductive productivity* pattern is defined according to the attributes that define a pattern:

Table 2. Example Patterns in GSD

Attribute	Description
Pattern name	***Unproductive productivity***
Problem	Vendor-team members work on tasks that are not really necessary. Improved focus on tasks that matter will increase productivity. Tasks are considered productive when they produce the results that the client requires despite the vendor perception that it is unproductive.
Population	Client and vendor teams from different organizational cultures.
Analysis of the problem	Unproductive-productivity happens when two different cultural models of productivity appear: client team has a numerical perception, whereas vendor team perceives productivity from the perspective of completion and quality of the work.
Solution	Vendors must communicate their perception about the tasks assigned to include how they view productivity. Clients have to be flexible in their productivity indicators in order to consider vendor´s estimations and suggestions.
Reference	[13]

Both, cultural dimensions and linguistic patterns can be defined by means of the attributes, as in the example in Table 2. Moreover, patterns can be considered as antipatterns depending on the context (scenarios).

B. GSD Scenarios

Scenarios define specific situations that can happen during the software process lifecycle in GSD. The different processes followed, composition of the teams, human factors, project characteristics or collaboration difficulties can influence a particular scenario in a different way. For this reason, defining accurate GSD scenarios requires the specific context in which they happen to be made clear. The scenarios will be stored along with their associated context and may be associated with an instance of a pattern. Recording the scenario–pattern relationship in this way allows for transparency, traceability and repeatability.

Šmite et al. [19] developed a classification scheme for GSD-related empirical research analysis. We apply some of their classification attributes in the definition of the context in which GSD scenarios take place.

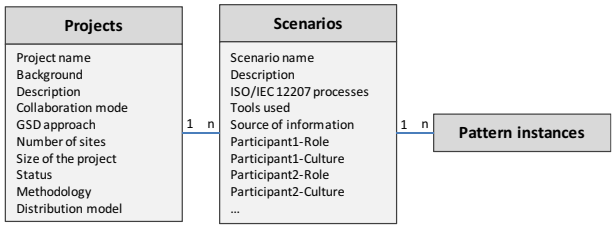

Figure 2. GSD scenario definition

The data model for storing the contextualized scenarios is set out in Figure 2, comprising information related to the project background in which the scenarios happen, along with the details of the scenario and the specific instances of patterns that appeared in it. The **project** entity contains the following attributes:

- Project name.
- Background: laboratory, industry.
- Description: details of the project.
- Collaboration mode: inter-organizational, intra-organizational.
- GSD approach: module-based, phase-based, follow the sun.
- Number of distributed sites.

- Size of the project: number of participants.
- Status of project: on time, delayed.
- Methodology applied: Ad-hoc, Waterfall, Prototype, Iterative, Extreme Programming (XP), Scrum, Agile.
- Distribution model: outsourcing, insourcing, offshoring, nearshoring, offshore outsourcing.

Each software development project can have several associated scenarios. For example; in the context of a project it is possible to consider a chat conversation among developers in an attempt to solve a problem, or an email conversation between project manager and developers for the assigning of tasks. The **scenario** entity covers the following information:

- Scenario name.
- Description: details of the scenario.
- Software processes related to the context of the problem.
- Source of information: origin and method of empirical data collection: audio recordings (meeting, telephone), case study, existing communities or forums on the field of GSD, experience, experiment, interviews, literature in the field of GSD, logs of conversations (e-mail, chat), pedagogical materials, problems reported by companies, observations, reports and surveys.
- Tools used in the scenario.
- Roles, responsibilities and cultures of the stakeholders involved.

When a new scenario is created, and depending on the particular definition of the GSD scenario, several GSD patterns can also be instantiated and associated with the scenario. The attributes that define an instance of a pattern depend on the kind of pattern. Finally, pattern instances and scenarios can have attachments providing specific information that could be analyzed during the review process detailed in Section VI.

IV. GSD COMMUNITY WEB

A collaborative website (http://global.lero.ie/community) has been designed as a means to share, find and discuss patterns and scenarios that happen in GSD. News, discussions and GSD-related resources are also considered in the quest to promote sharing of information and ideas. Instructors, researchers and practitioners with knowledge on GSD are the potential collaborators and users.

Project Background

Project name	ORIGIM
Background	Industry
Description	
Collaboration mode	Intra-organizational
Distributed sites involved	4
GSD approach	Module Based
Size of the project (people involved)	15
Status of the project	On time
Methodology	Agile

Scenario

Project	ORIGIM
Scenario name	Requirements Elicitation
Description	
ISO/IEC 12207 processes	Acceptance
Tools used	Eclipse
Source of information	Audio recordings (meeting, telephone)
Participant 1 - Role	Analyst
Participant 1 - Culture	Spain
Participant 2 - Role	Customer
Participant 2 - Culture	Portugal

Figure 3. Definition of projects and scenarios

New participants can register in the community so that they can share their knowledge and participate in forum discussions. They can also propose improvements, share their problems or needs, and propose surveys or new features for specific purposes of interest to them and for the common good of the community. Figure 3 shows the definition of the background of a GSD project (right) and one of its associated scenarios (left) in which a Spanish analyst is interacting with a Portuguese customer.

Figure 4 shows instances of cultural patterns which represent real conversations in the context of the aforementioned scenario. The site also includes features for sorting and grouping information by taking the different attributes into account, exporting information, version control management, search engine, workflow support and automatic notifications.

Figure 4. Pattern instances

V. KNOWLEDGE ACQUISITION PROCEDURE

Members of the community can collaborate by including their scenarios and associated patterns, based on real problems and experiences on GSD. The process for gathering this information by means of the website is set out in Figure 5; it is made up of the following steps (A, B, C):

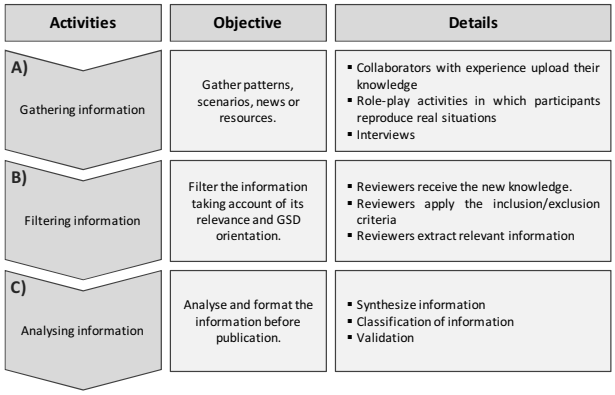

Figure 5. Knowledge acquisition process

A) Gathering information

In order to populate the knowledge base, three methods are considered:

- Interested participants (researchers or practitioners) actively provide their knowledge or experience that could be translated into patterns or scenarios.

- Using a role-play activity in which groups from several organizations are asked to act through their typical work. As they do so, they capture information on Class Responsibility Collaboration (CRC) cards that can be analyzed to identify patterns and scenarios. This method is similar to the one followed by Coplien and Harrison [8] to obtain Organizational Patterns.
- Interviews with experts in which they will be asked to provide their experiences in a structured manner.

B) Filtering information

- The submission of new material is automatically notified to a group of reviewers.
- Reviewers examine the new submission, checking that it is consistent with the objectives and that there are no similar cases already stored in the knowledge base (by means of the search engine). Then reviewers apply the following inclusion and exclusion criteria:
  Inclusion criteria: Problems that can appear in a GSD context and can be generalized and applied in similar settings.
  Exclusion criteria: Non-representative problems, issues that depend on personality factors or conflicts that has not come about as a consequence of applying GSD.
- Extract relevant information. Reviewers can modify the content by extracting relevant information.

C) Analyzing information

- Synthesize information. Data is formatted and useless or repetitive information is removed.
- Classification of information. This phase involves checking the correctness of the initial data classification.
- Validation. This phase validates the information and makes it available to the community.

Finally, after publication on the web, all forms of information can also be commented on and updated.

VI. DISCUSSION

Because GSD involves social factors and communication difficulties, as well as language and cultural distances, GSD research tends to be criticized for being too closely-related to the field of sociology, rather than dealing purely with Software Engineering technical issues. Sometimes it is not easy for researchers to define the separation between GSD and sociology. Having a set of recognized GSD patterns (and antipatterns) could become part of a framework or ontology of specific areas of concern in GSD that cut across the socio-technical divide.

Identification and classification of different levels of GSD patterns and concrete scenarios have other potential benefits for the GSD community. First of all, it provides a way of sharing and discovering challenges or problems that members of the community could see as interesting for their own study. Secondly, the potential reuse of the patterns and scenarios provides the opportunity for the community to act as a test bed for evaluating and testing the patterns and scenarios, as well as for improving them iteratively. Finally, the knowledge generated can be applied for industrial, educational and research purposes.

One of the concerns in the design of the definition of the patterns and scenarios was to minimize collaborator effort. The classification of the information and the search engine help users to find relevant information. The possibility of contributing with raw data that would be revised afterwards by the reviewers was considered in the knowledge acquisition process. Contextualization of information and reuse of existing information is also intended to reduce this effort.

An important limitation for obtaining a representative knowledge base is the willingness of the community to participate and share their experiences. In order to tackle this problem, our future work will consider gathering training scenarios through conducting confidential interviews with practitioners and also through the use of role-play. Role play was found to be a successful method for identifying scenarios and patterns in Coplien and Harrison [8].

VII. CONCLUSIONS

In this paper a pattern model for GSD is presented, with the aim of providing a knowledge base that could be useful for the GSD community. Patterns can be used to apply specific protocols or solutions for solving specific scenarios. Apart from that, scenarios help to understand the patterns and their context of applicability. The aim of this model is to break a problem down into simpler ones, to package experiences for reuse, and make them available for studying real problems. The objective is to use this knowledge for research and educational purposes; it also serves as a tool that allows researchers to place their work within a framework of patterns that are recognised by the GSD community.

The next step of this work is to create a community of researchers and practitioners interested in sharing their knowledge, or wanting access to the information. To achieve this, we will invite companies and authors of relevant papers in the GSD literature to participate in this knowledge-sharing. The advantages of this model are:

- Flexibility in adapting the pattern model, with new entities and attributes being included.
- Availability of knowledge to the general public and participants, fostering collaboration.
- Easy to learn for collaborators. It promotes discussion and a participatory research paradigm.
- Easy access to results, which can be summarized or grouped according to different criteria, thus promoting analysis and reuse.

Efforts will be made to extract new problems and scenarios and contextualize them in the GSD field, making them available to the community. Finally, the structure of the repository has to be validated by GSD experts and the resulting knowledge should be applied in academia and industry. Future work will focus on defining critical success factors that are considered essential in the assessment of the patterns and scenarios. We intend to study the feasibility of

applying an adapted version of the evaluation technique Q-PAM [20]. The general idea of Q-PAM is to use scenarios as test cases which are analyzed against the patterns. These patterns and scenarios can also be validated by proving their effectiveness when they are applied in training environments to develop specific skills.

ACKNOWLEDGMENTS

This work was supported, in part, by the Science Foundation Ireland grant 10/CE/I1855 to Lero - the Irish Software Engineering Research Centre (www.lero.ie). It has also been funded by the GEODAS-BC project (Ministerio de Economía y Competitividad and Fondo Europeo de Desarrollo Regional FEDER, TIN2012-37493-C03-01). Additional support came from ORIGIN (IDI-2010043 (1-5)) funded by CDTI and FEDER, as well as GLOBALIA (PEII11-0291-5274), Consejería de Educación y Ciencia, Junta de Comunidades de Castilla-La Mancha.

REFERENCES

[1] M. J. Monasor, M. Piattini, and A. Vizcaíno, "Challenges and Improvements in Distributed Software Development: A Systematic Review," *Advances in Software Engineering,* vol. 2009, pp. 1-16, 2009.

[2] M. J. Monasor, A. Vizcaíno, and M. Piattini, "Cultural and linguistic problems in GSD: a simulator to train engineers in these issues," *Journal of Software Maintenance and Evolution: Research and Practice (Special Issue on Global Software Engineering),* vol. 24, pp. 707-717, 18 Aug 2011.

[3] E. Carmel, *Global Software Teams: Collaborating Across Borders and Time Zones*. Upper Saddle River, NJ, USA: Prentice Hall, 1999.

[4] E. Hargreaves, D. Damian, F. Lanubile, and J. Chisan, "Global software development: building a research community," *SIGSOFT Softw. Eng. Notes,* vol. 29, pp. 1-5, 2004.

[5] K. Karlgren and R. Ramberg, "The Use of Design Patterns in Overcoming Misunderstandings in Collaborative Interaction Design," *CoDesign - International Journal of CoCreation in Design and the Arts,* vol. 8, pp. 231-246, 2012.

[6] C. Alexander, *A pattern language: towns, buildings, construction* vol. 2: Oxford University Press, USA, 1977.

[7] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns CD: Elements of Reusable Object-Oriented Software* vol. 1: Addison-Wesley Professional, 1998.

[8] J. O. Coplien and N. B. Harrison, *Organizational Patterns of Agile Software Development*: Prentice Hall, 2005.

[9] J. Bergin, J. Eckstein, M. L. Manns, H. Sharp, J. Chandler, K. Marquardt, E. Wallingford, M. Sipos, and M. Völter, *Pedagogical Patterns: Advice for Educators*: Software Tools, 2012.

[10] I. M. Jorrín-Abellán, I. Ruiz-Requies, D. Hernández Leo, E. D. Villasclaras-Fernández, Y. Dimitriadis, B. Rubia-Avi, and J. I. Asensio-Pérez, "Collage, a Collaborative Learning Design Editor Based on Patterns," *Educational Technology & Society,* vol. 9, pp. 58-71, 2006.

[11] P. A. Laplante and C. J. Neill, *AntiPatterns: Identification, Refactoring, and Management*: Auerbach Publications, 2006.

[12] E. MacGregor, Y. Hsieh, and P. Kruchten, "Cultural patterns in software process mishaps: incidents in global projects," in *Proceedings of the 2005 workshop on Human and social factors of software engineering* St. Louis, Missouri: ACM, 2005, pp. 1-5.

[13] H. Shah, N. J. Nersessian, M. J. Harrold, and W. Newstetter, "Studying the influence of culture in global software engineering: thinking in terms of cultural models," in *Proceedings of the 4th international conference on Intercultural Collaboration* Bengaluru, India: ACM, 2012.

[14] M. Paasivaara, C. Lassenius, and P. Pyysiainen, "Communication Patterns and Practices in Software Development Networks," Helsinki, Finland: Helsinki University of Technology, 2003.

[15] A. Välimäki, J. Kääriäinen, and K. Koskimies, "Global Software Development Patterns for Project Management," in *Software Process Improvement*. vol. 42: Springer Berlin Heidelberg, 2009, pp. 137-148.

[16] A. Pehmöller, F. S. Capgemini, and S. Wagner, "Patterns for testing in global software development," in *13th International Conference on Quality Engineering in Software Technology*, 2010.

[17] F. Salger, J. Englert, and G. Engels, "Towards Specification Patterns for Global Software Development Projects - Experiences from the Industry," in *Quality of Information and Communications Technology (QUATIC), 2010 Seventh International Conference on the*, pp. 73-78.

[18] G. Hofstede and G. J. Hofstede, *Cultures and organizations: software of the mind*, 2nd ed. New York, NY, USA, 2005.

[19] D. Smite, C. Wohlin, R. Feldt, and T. Gorschek, "Reporting Empirical Research in Global Software Engineering: A Classification Scheme," in *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, 2008, pp. 173-181.

[20] A. Välimäki, S. Vesiluoma, and K. Koskimies, "Scenario-Based Assessment of Process Pattern Languages," in *Product-Focused Software Process Improvement*. vol. 32: Springer Berlin Heidelberg, 2009, pp. 246-260.