European ystems & oftware rocess mprovement and nnovation

16th EuroSPI² Conference

Industrial Proceedings

2 - 4 September 2009

University of Alcalá. Alcalá de Henares. Spain

http://www.eurospi.net

ISBN 978-87-7398-151-1

Publisher: DELTA. http://www.delta.dk

Printer: PUBLIZON. http://www.publizon.dk

Partnership:

ASQF. http://www.asq.org
DELTA. http://www.delta.dk
ISCN. http://www.iscn.com
ISQI. http://www.isqi.de
SINTEF. http://www.sintef.no
FiSMA. http://www.fisma.fi

Supporter:

University of Alcalá.
http://www.uah.es

EuroSPI 2009 Proceedings

Proceedings

The papers in this book comprise the industrial proceedings of the EuroSPI 2009 conference. They reflect the authors' opinions and, in the interests of timely dissemination, are published as presented and without change.

Their inclusion in this publication does not necessarily constitute endorsement by EuroSPI and the publisher.

DELTA Series about Process Improvement – ISBN 978-87-7398-151-1.

EuroSPI

EuroSPI is a partnership of large Scandinavian research companies and experience networks (SINTEF, DELTA, STTF), the ASQF as a large German quality association, the American Society for Quality, and ISCN as the co-ordinating partner.

EuroSPI conferences present and discuss practical results from improvement projects in industry, focussing on the benefits gained and the criteria for success. Leading European industry are contributing to and participating in this event. This year's event is the 16th of a series of conferences to which countries across Europe and from the rest of the world contributed their lessons learned and shared their knowledge to reach the next higher level of software management professionalism.

EuroSPI Chairs

General Chair	Dr Richard Messnarz, (ISCN, Ireland)
EuroSPI Marketing Chair	Prof. Miklos Biro (Corvinus University, Hungary)
Scientific Program Chair	Prof. Juan Cuadrado Gallego (University of Alcala, Spain)
Scientific Program Chair	Dr Ricardo Rejas Muslera (University of Alcala, Spain)
Scientific Program Chair	Dr Rory O Connor (Dublin City University, Ireland)
Scientific Program Chair	Prof. Kari Smolander (Lappeenranta University of Tech. Finland)
Scientific Program Chair	Dr Nathan Baddoo (University of Hertfordshire, UK)
Industrial Program Chair	Risto Nevalainen FiSMA and STTF
Industrial Program Chair	Stephan Goericke, ISQI
Industrial Program Chair	Jorn Johansen, DELTA
Industrial Program Chair	Mads Christiansen, DELTA
Industrial Program Chair	Nils Brede Moe, SINTEF
Tutorial Chair	Dr Richard Messnarz, ISCN
Organizing Chair	Adrienne Clarke BA, ISCN

Industrial Program Committee

BACHMANN Volker SIBAC GmbH GERMANY
BALSTRUP Bo Center for Software Innovation DENMARK
BARAFORT Beatrix Centre de Recherche Public Henri Tudor LUXEMBOURG
BRESKE Eva Robert Bosch GmbH GERMANY
CHRISTIANSEN Mads DELTA DENMARK
COLLINO Alessandro ONION S.p.A. ITALY
COLOMO PALACIOS Ricardo Universidad Carlos III de Madrid SPAIN
DAUGHTREY Taz H. James Madison University USA
DAVIES Richard Schlumberger WesternGeco NORWAY
DUSSA-ZIEGER Claudia Method Park Software AG GERMANY
EKERT Damjan ISCN AUSTRIA
FEHRER Detlef SICK AG GERMANY
HAGENMEYER Philipp ZF Friedrichshafen AG GERMANY
HIND Tim Axa Sun Life UK
JOHANSSON Mika Finnish Software Measurement Association FiSMA FINLAND
KASTNER Norbert sepp.med gmbh GERMANY
LICHTENECKER Gerhard MAGNA STEYR Fahrzeugtechnik AUSTRIA
MESSNARZ Richard ISCN IRELAND
MORGENSTERN Jens TUEV GERMANY
NEVALAINEN Risto Falcon Leader Oy FINLAND
O LEARY Eugene EQN Ltd. IRELAND
PIKKARAINEN Mika VTT Technical Research Centre of Finland FINLAND
POTH Alexander SQS Software Quality Systems AG GERMANY
RENAULT Samuel Centre de Recherche Public Henri Tudor LUXEMBOURG
RIEL Andreas Grenoble Institute of Technology FRANCE
ROMCEA Cristina Conti Temic microelectronic GmbH GERMANY
SANDERS Marty UK
SCHWEIGERT Tomas SQS Software Quality Systems AG GERMANY
SEHL Georg iMBUS AG GERMANY
SMITE Darja Blekinge Inst. of Technology SWEDEN
SPORK Gunther MAGNA Powertrain AUSTRIA
STEFANOVA-PAVLOVA Maria Center for Innovation and Technology Transfer-Global BULGARIA
ULSUND Tor Geomatikk AS NORWAY
VON BRONK Peter Systemberatung Software-Qualität GERMANY
WÖRAN Bruno DANUBE AUSTRIA

EuroSPI Board Members

ASQ, http://www.asq.org
ASQF, http://www.asqf.de
DELTA, http://www.delta.dk
ISCN, http://www.iscn.com
SINTEF, http://www.sintef.no
FiSMA, http://www.fisma.fi

Editors of the Proceedings

Richard Messnarz, ISCN, Ireland
Sonja Koinig, ISCN, Austria
Mads Christiansen, DELTA, Denmark
Jorn Johansen, DELTA, Denmark
Juan Cuadrado Gallego, University of Alcala, Spain

Welcome Address by the EuroSPI General Chair

Dr Richard Messnarz

EuroSPI is an initiative with 4 major goals (www.eurospi.net):

1. An annual EuroSPI conference supported by Software Process Improvement Networks from different EU countries.
2. Establishing an Internet based knowledge library, newsletters, and a set of proceedings and recommended books.
3. Establishing an effective team of national representatives (in future from each EU country) growing step by step into more countries of Europe.
4. Establishing a European Qualification Framework for a pool of professions related with SPI and management. This is supported by European certificates, exam systems, and online training platforms (www.ecqa.org).

EuroSPI is a partnership of large Scandinavian research companies and experience networks (SINTEF, DELTA, STTF), the ASQF as a large German quality association, the American Society for Quality, and ISCN as the co-coordinating partner. EuroSPI collaborates with a large number of SPINs (Software Process Improvement Network) in Europe.

EuroSPI conferences present and discuss results from software process improvement (SPI) projects in industry and research, focusing on the benefits gained and the criteria for success. Leading European universities, research centers, and industry are contributing to and participating in this event. This year's event is the 16th of a series of conferences to which international researchers contribute their lessons learned and share their knowledge as they work towards the next higher level of software management professionalism.

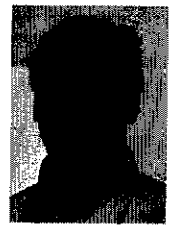
The greatest value of EuroSPI lies in its function as a European knowledge and experience exchange mechanism for Software Process Improvement and Innovation of successful software product and service development. EuroSPI aims at forming an exciting forum where researchers, industrial managers and professionals meet to exchange experiences and ideas and fertilize the grounds for new developments and improvements.

EuroSPI also established an umbrella initiative for establishing a European Qualification Network in which different SPINs and national initiatives join mutually beneficial collaborations (EU Certificates Campus www.eu-certificates.org, European Certification and Qualification Association www.ecqa.org).

A cluster of European projects (supporting ECQA and EuroSPI) contribute knowledge to the initiative, including currently EU Cert (EU Certificates Campus), iDesigner (integrated mechatronics designer), MONTIFIC (Financial SPICE Assessor), ELM (e-learning manager), CROMEU (EU Managers in South eastern Europe), etc. A pool of more than 20 qualifications has been set up.

Please join the community of cross company learning of good practices!

Welcome to Spain by Prof. Juan Cuadrado Gallego and Dr Ricardo Rejas Muslera



Prof. Juan Cuadrado Gallego, University of Alcala, Spain

Dr Ricardo Rejas Muslera, University of Alcala, Spain

The University of Alcalá (Spanish: Universidad de Alcalá) is a public university located in the city of Alcalá de Henares, to the east of Madrid in Spain. Founded in 1499, it was moved in 1836 to Madrid. In 1977, the University was reopened in its same historical buildings. The University of Alcalá is especially renowned in the Spanish-speaking world as it presents each year the highly prestigious Cervantes Prize.

Today's University of Alcalá preserves its traditional humanities faculties, a testimony to the university's special efforts, past and present, to promote and diffuse the Spanish language through both its studies and the Cervantes Prize, which is awarded annually by the King and Queen of Spain in the Paraninfo (Great Hall). The University has added to its time-honoured education in the humanities and social sciences new degree subjects in scientific fields such as health sciences or engineering, spread out across its different sites (the Alcalá Campus, El Encín, and Guadalajara), all of which, together with the Science and Technology Park, are a key factor in its projection abroad, while also acting as a dynamo for activities in its local region.

Prof. Juan Cuadrado Gallego is a well known expert in the field of measurement and mathematical models used to measure processes and products. He has opublished numerous scientific papers in this field.
Dr Ricardo Rejas Muslera has done extensive research work in the use of assessment and improvement models for legal aspects, so called legal assurance processes.

We are welcoming EuroSPI 2009 to Alcala, a historic place with one of the oldest universities from Spain. University of Alacala is a member of international research networks and actively contributes process and product measurement knowledge to the European SPI community. We believe that this joint conference will fertilize the grounds for empowering networks and partnerships in SPI between the Spanish and the European wide communities.

Please join our Spanish SPI community and lets create connections Europe wide!

Welcome by DELTA, Editors of the DELTA Improvement Series

DELTA has been working with Software Process Improvement (SPI) for more than 15 years including maturity assessment according to BOOTSTRAP, SPICE and CMMI. DELTA has also been a partner in the EuroSPI conference from the very beginning 16 years ago. We are now for the 2.nd. time the publisher of the Industrial Proceedings from EuroSPI making it part of the DELTA series about Process Improvement.

Jørn Johansen, DELTA, Denmark

Jørn Johansen is Manager of the DELTA Axiom department at DELTA. He has an M.Sc.E.E. from Ålborg University and more than 28 years experience in IT. He has worked in a Danish company with embedded and application software as a Developer and Project Manager for 15 years. Mr. Johansen has been involved in all aspects of software development: specification, analysis, design, coding, and quality assurance. Furthermore he has been involved in the company's implementation of an ISO 9001 Quality System and was educated to and functioned as Internal Auditor.

For the last 14 years he has worked at DELTA as a consultant and registered BOOTSTRAP, ISO 15504 Lead Assessor, CMMI Assessor and *ImprovAbility*™ Assessor. He has participated in more than 40 assessments in Denmark and abroad for companies of all sizes. He was the Project Manager in the Danish Centre for Software Process Improvement project, a more than 25 person-year SPI project and Talent@IT, a 26 person-year project that involves 4 companies as well as the IT University in Copenhagen and DELTA. Currently Mr. Johansen is the Project Manager of SourceIT an 18 person-year project focusing on outsourcing and maturity. Mr. Johansen is also the co-ordinator of a Danish knowledge exchange group: Improving the Software Development Process, which is the Danish SPIN-group.

Contact: Jørn Johansen, DELTA, Venlighedsvej 4, DK 2970 Hørsholm, Denmark, phone +45 72 19 40 00 or e-mail : joj@delta.dk

Mads Christiansen, DELTA, Denmark

Mads Christiansen has an M.Sc.E.E. from DTU (Danish Technical University) and more than 30 years experience in product development and IT. He has worked for 19 years in a Danish company with embedded and application software as a Developer and Project Manager. Mr. Christiansen has been involved in all aspects of software development: specification, analysis, design, coding, and quality assurance and managing outsourced projects in Denmark and USA.

For the last 11 years he has worked at DELTA as a consultant in SPI (requirements specification, test, design of usable products and development models). Currently Mr. Christiansen works with eBusiness and as Innovation Agent. Mr. Christiansen is also *ImprovAbility*™ Assessor and Trainer of *ImprovAbility*™ project Assessors.

Contact: Mads Christiansen, DELTA, Venlighedsvej 4, DK 2970 Hørsholm, Denmark, phone +45 72 19 40 00 or e-mail : mc@delta.dk

Contents

Experience Session I: SPI and Safety Engineering

Additional Requirements for Process Assessment in Safety-Critical Software and Systems Domain 1.1
Mika Johansson, Risto Nevalainen

What is a Test Case? Revisiting the Software Test Case Concept 1.13
Dani Almog, Tsipi Heart

Addressing IEC 61508 Certification in a multi-standards context: A generic approach 1.27
Gautier Dallons, Annick Majchrowski, Jean-Christophe Trigaux

Experience Session 2: SPI and Systems Engineering

Improving the Software-Development for multiple Projects by applying a Platform Strategy for Mechatronic Systems 2.1
Volker Bachmann, Richard Messnarz

Integrated Engineering Collaboration Skills to Drive Product Quality and Innovation 2.11
Andreas Riel, Serge Tichkiewitch, Anca Draghici, George Draghici, Damian Grajewski, Zenobia Weiss, Richard Messnarz

A Systems Approach to Software Process Improvement in Small Organisations 2.21
Diana Kirk and Stephen MacDonell

Experience Session 3: SPI and Knowledge Engineering

SPI Based Learning Organisations 3.1
Richard Messnarz, Gearoid O'Suilleabhain, Damjan Ekert

The SPI Professional Qualification Scheme 3.13
Tomas Schweigert, Richard Messnarz, Morten Korsaa, Jorn Johansen, Risto Nevalainen, Miklos Biro

Methodology for Process Improvement through Basic Components and Focusing on the Resistance to Change 3.23
Calvo-Manzano Jose A., Cuevas Gonzalo, Gómez Gerzon, Mejia Jezreel, Muñoz Mirna, San Feliu Tomás, Sánchez Angel

A Dynamic Model for Improving Knowledge Sharing in Virtual Organisations 3.37
Elli Georgiadou, Kerstin V. Siakas

Experience Session 4: SPI Implementation Experiences

CMMI based Continuous Improvement Program in a financial software company – Results and Experiences 4.1
Erkki Savioja, Risto Nevalainen

Lessons learned from an ISO/IEC 15504 SPI Programme in a Company 4.13
Antonia Mas, Bartomeu Fluxà, Esperança Amengual

Improving software processes using CMMI in a pragmatic way: experience at ING Luxembourg 4.19
Valérie Betry, Stéphane Cortina, Séverine Mignon

Two Years Defining and Implementing Software Processes: The Experience of a Brazilian University 4.29
Adriano Bessa Albuquerque, Nabor Chagas Mendonça, Carla Ilane Moreira Bezerra, Shamya Ribeiro Gomes, Alexandre Lima Gondim

Experience Session 5: SPI and Testing

Combining EXAM with model-centric testing 5.1
Anne Kramer

Make Test Process Assessment similar to Software Process Assessment – The Spice 4 Test approach 5.9
Monique Blaschke, Michael Philipp, Tomas Schweigert

A Model to Select Testable Use Cases: a Real Experience in a Financial Institution 5.19
Silva Andrea Rodrigues, Rodrigues Maikol Magalhães, Pinheiro Plácido Rogério, Albuquerque Adriano Bessa, Gonçalves Francisca Márcia

Test-Driven Automation: Adopting Test-First Development to Improve Automation Systems Engineering Processes 5.25
Dietmar Winkler, Stefan Biffl, Thomas Östreicher

Experience Session 6: SPI and Assessment Models

Processes Implementation Sequences to Achieve the Maturity Level 2 in CMMI-ACQ 6.1
Calvo-Manzano Jose A., Cuevas Gonzalo, Gómez Gerzon, Mejia Jezreel, Muñoz Mirna, San Feliu Tomás, Sánchez Angel

Development of a concept for integrating various Quality Standards 6.13
Damjan Ekert, Philipp Hagenmeyer, Richard Messnarz

A Method Framework for Engineering Process Capability Models 6.25
Clenio F. Salviano, Alessandra Zoucas, Jorge V. L. Silva, Angela M. Alves, Christiane G. von Wangenheim, and Marcello Thiry

Analysis of the Design of Software Process Assessment Methods from an Engineering Design Perspective 6.37
Mohammad Zarour, Alain Abran, Jean-Marc Desharnais

Experience Session 7: SPI and Measurement

Experiment with COSMIC V3.0: Case Studies in Business Applications 7.1
Sanae Saadaoui, Annick Majchrowski, Christophe Ponsard

A Review of Effort Estimation Lineal Mathematical Models for the Software Project Planning Process 7.11
Borja Martín, Pablo R. Soria, María-José Domínguez, Marian Fernández, José-Luis Cuadrado

Measurement and Quantification are Not the Same: ISO 15939 and ISO 9126 7.21
Alain Abran, Jean-Marc Desharnais, Juan Jose Cuadrado-Gallego

Experience Session 8: SPI and Agile

Towards Agile Product Derivation in Software Product Line Engineering 8.1
Padraig O'Leary, Fergal McCaffery, Ita Richardson, Steffen Thiel

Being Agile when Developing Software Components and Component-Based Systems – Experiences from Industry 8.7
Iva Krasteva, Rikard Land, A. S. M. Sajeev

Agile Software Development in Distributed Environments 8.19
Kerstin V. Siakas, Errikos Siakas

Experience Session 9: Utilising Improvement Models

A CMMI Ontology for an Ontology-Based Software Process Assessment Tool 9.1
Sema Gazel, Aya Tarhan, Ebru Sezer

Utilizing a Metrics Knowledge Base for Software Process Improvement 9.9
Timo Mäkinen, Jari Soini, Timo Varkoi

Software process improvement within a software R&D department 9.17
M. Kevitt & R. Verbruggen

Experience Session 10: SPI and IT Services

TIPA to keep ITIL going and going 10.1
Marc St-Jean

Process improvement in IT departments: a four years experience in a large retail company 10.9
Paolo Salvaneschi, Daniele Grasso, Maurizio Besurga

Service Level Management for IT Services in small settings: a Systematic Review 10.19
Calvo-Manzano Jose, Cuevas Gonzalo, Gómez Gerzon, Mejia Jezreel, Muñoz Mirna, Rabbi Forhad, San Feliu Tomás

Experience Session 11: SPI and SW Engineering

Integration of CASE Tools to Software Processes: A Case Study 11.1
K. Alpay Erturkmen, Onur Demirors

Experiences of teaching code reviews for IT students 11.13
Zádor Dániel Kelemen, Katalin Balla

Experience Session 12: SPI and Processes

Tool Support for Collaborative Software Process Authoring in Large Organizations 12.1
Jeanette Heidenberg, Petter Holmström, Ivan Porres

An Enhanced Variability Mechanisms to Manage Software Process Lines 12.13
Tomás Martínez-Ruiz, Félix García, Mario Piattini

Survey of Practitioners' Attitudes to Software Process Modelling 12.25
Tuomas Mäkilä, Henrik Terävä

Experience Session 13: SPI in SMEs

Approaching a software factory process philosophy for a SME in Spain 13.1
Luis Fernández-Sanz, José R. Hilera, José J. Martínez, Guillermo López, Jesús Serrato

Successful Process Improvement for Small and Medium Sized set ups 13.9
Kurt S. Frederichsen

Defining and Implementing Software Subcontracting Management Processes in Small Enterprises 13.21
Calvo-Manzano J. A., García, I., Pacheco C., Sumano P.

Additional Requirements for Process Assessment in Safety-Critical Software and Systems Domain

Mika Johansson [1], Risto Nevalainen[2]

[1] Tampere University, Pori Unit, Finland
and FiSMA ry, Finland
[2] FiSMA ry, Finland

Abstract

Certification of safety-critical software is a multi-disciplinary topic. Process assessment is an essential part of that, but is not enough for software certification. Certification employs also several other method families, like inspections and reviews, independent V&V, conformance with selected reference standard(s) and use of selected measurements and analyses.

Process assessment supports directly qualification of safety-critical applications but is less relevant for certification of platforms and environments. Anyway, qualification and certification are closely related, because certification as a whole supports qualification and makes it more effective. It is possible to adapt and evolve process assessment so, that it supports both qualification and certification.

Typical process assessment is done for improvement purpose. In qualification and certification that is not so relevant as conformance and management of risks. In this paper we discuss about possibilities to develop process assessment to achieve that goal. In most cases assessment is a combination of several approaches.

Keywords

Safety-critical software, process assessment, conformance with standards

An Enhanced Variability Mechanisms to Manage Software Process Lines

Tomás Martínez-Ruiz, Félix García, Mario Piattini
Alarcos Research Group, Information and Technology Systems Department
Escuela Superior de Informática, University of Castilla-La Mancha
Paseo de la Universidad, 4, 13071 Ciudad Real, Spain
{tomas.martinez, felix.garcia, mario.piattini}@uclm.es

Abstract

The characteristics of projects and organizations influence process development and must therefore be taken into account during process tailoring. Software process lines permit process adaptation to include these characteristics in processes. However, it is necessary to define suitable variability mechanisms for software processes, considering the elements that take part in them, their relationships and their evolution and improvement. An analysis of processes, the elements of which they are composed, and how they may vary, permit the variability mechanism used in product lines to be applied and enhanced to tailor software processes. Different types of variants and variation points can be distinguished according their use in tailoring processes. These can be used to obtain tailored processes by paying attention to the elements which will be varied, since the others are managed automatically, thus providing the means for them to be used through their improvement. The proposed mechanisms consider simplified tailored process creation whilst simultaneously ensuring transparent process consistence from the process engineer's point of view.

Keywords

Software Process Lines, Software Process Tailoring, Variability Mechanisms, Variant, Variation Point, Dependences, Software Process Institutionalization

1 Introduction

Software development has evolved from artisan methods to become more engineering centred, resulting in the creation of software product lines [1]. This approach permits the management of families of products, which offers some advantages. On the one hand, software product lines make it possible to tailor products to the user's needs by making good use of both the similarities and differences between the products. On the other hand, the costs of creating products can be reduced, and the quality of the product line and its products can be improved while they are generated [1].

Bearing in mind that "software process are process too" [2], several software engineering techniques have been applied in software process, and this is termed as process engineering. Software process lines are thus created from product lines [3]. The use of this approach permits processes to be tailored to the characteristics of both projects (since "just as there are no two identical projects, there are no two identical processes in the world" [4]) and organizations [5], which is necessary for their survival [6]. Moreover, some process models, such as CMMI, specify software process improvement, which implies that processes need to be engrained in the way in which work is carried out in organizations [7]. In order to engrain, these processes must be tailored according the specific characteristics of each project and organization. The inclusion of organization characteristics in generic processes by means of tailoring is thus a key factor in engraining processes in organizations [8].

It is also necessary to learn about the adapted processes in order to improve the way in which the processes are tailored, and therefore improve the tailored process. Supporting the adaptation of processes by using well defined and well bounded variations implies that any changes are well-known and well controlled, which facilitates learning about their realization and how to improve it [8]. The mechanisms which permit variations are variants and variation points [9]. The former represent single parts that vary, and the latter are the parts of the process into which variants can be inserted. They also define dependences in order to carry out difficult tasks such as guaranteeing the consistence of the generated processes.

However, in order to ensure ease of consistence and to make its application possible by means of automatic tools, both a clear specification of the variability elements and a description of their behavior and the definition of the scenarios in which they may interact are necessary. Tailored processes can thus be simplified to select the variants which are most suitable for the organization and project, knowing that the process is always consistent.

In this paper we have analyzed the various dependences that may be present in a process line, and how these dependences affect such variants or variation points. This analysis has allowed us to classify both variants and variation points according to their behaviour and, what is more important, according to how the process engineer views them when s/he is about to tailor a process. Furthermore, both the view of process lines and the way in which consistence is ensured can be simplified.

Section 2 presents the state-of-the-art in variability in process and product lines and a summarization of SPEM. Section 3 summarizes the previous approach to manage variability. Section 4 presents the analysis carried out, which is illustrated through an example. Finally, our conclusions and future work are presented in Section 5.

2 State of the art

Software process lines are a new discipline and, to the best of our knowledge, very few works concerning variability in process lines therefore exist. In [9] a variability approach to process lines is presented, which will be summarized in Section 3. Furthermore, the Fraunhofer IESE is working on variability in processes in order to customize software inspection, showing mechanisms based on variants over several process elements [10].

Several works concerning variability in workflows and business processes also exist. [11] describes how to use variability in software processes, since organizations must adapt their processes to survive

[6], in order to manage them as a process line [12]. Variability must be based on conceptual processes, in a similar manner to that of product lines [13]. According to [14] variability allows the propagation of best practices.

However, as process lines are based on product lines, it is possible to apply the software product line variability mechanisms proposed in literature to the modelling of variability in software process lines. A variation point is the place in which variability occurs [15], and it has the advantage of adding new variability implementations, by adding new variants [16], the concrete elements that are placed at the variation points, each of which implements this variability in a different way [17]. Both of them can be distinguish by using stereotypes [18]. According to [19], it is possible to specify dependencies and constraints between these elements. In [20] three abstraction levels for modelling variability with dependencies are presented, and [21] presents patterns with which to build, manage and manipulate variation points.

If process lines are to be modelled in a usable way, they must be based on process models. SPEM (Software Process Engineering Metamodel) [22] is the Object Management Group (OMG) proposal through which to create software processes by using methods. This proposal is one of those most used by industry because it is independent from any process model.

As is explained in [9], the new 2.0 version of the standard permits variability when methods are being modelled, or when method components are used to compose process activities. However, it does not, in itself, permit variability in processes. Furthermore, the mechanisms included in SPEM are not able to define the core process of the software process line, and the common elements of all the processes in the line cannot, therefore, be guaranteed. Finally, SPEM does not include specific variability notation to facilitate the use of variability mechanisms when processes are to be tailored. An extension of SPEM is thus proposed below.

3 Variability in Software Process Lines

An extension of SPEM which permits processes to vary is proposed in [9]. All the elements of this proposal can be included in a new package in the SPEM *ProcessLineComponents* metamodel(Figure 1). This new package uses elements from the SPEM *ProcessWithMethods* and *ProcessStructure* subpackages, with the aim of adding the variation capability to the process constructors in both packages.

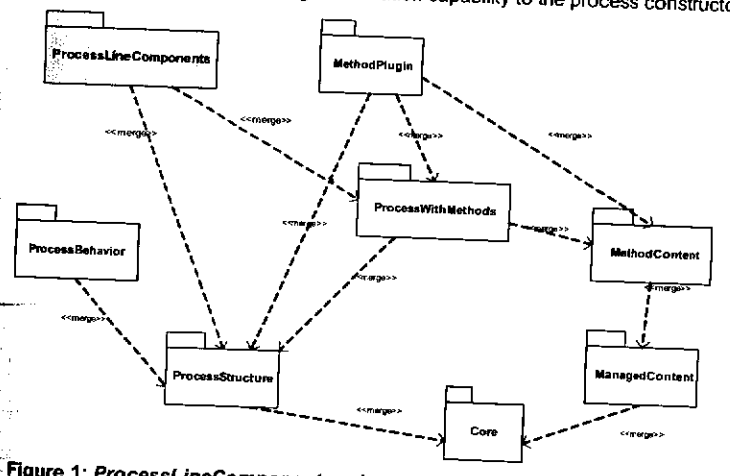


Figure 1: *ProcessLineComponents* sub-package in the SPEM metamodel

As is shown in [9], variability is basically managed by five elements. The first is the core process, which contains the common elements of all the processes of the process line with several empty

points, the variation points, in which the variation occurs. The variants are the concrete elements which are inserted into the variation points and tailor the resulting process. The *LProcElement* abstract class is a super-class of both elements, as Figure 2 shows. Variants and variation points are also abstract elements, and must be concretized from different SPEM concrete process elements as Figure 2 shows, by using the SPEM activity.

The other two elements are dependences between variants and variation points, and the occupation relationship, used to represent which variant is occupying which variation point. Occupation is a key element because it makes it possible to convert variation points into process elements, giving support to variability.

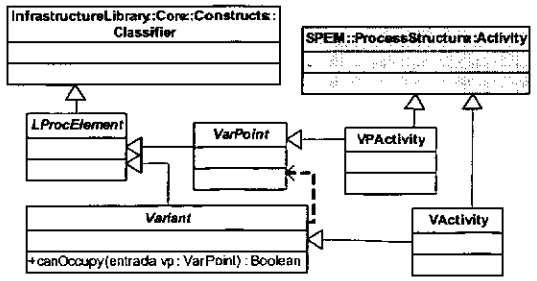


Figure 2: Metamodel of Variants and *VariationPoints* in the *ProcessLineComponents* package

Moreover, process elements such as activities, tasks, roles, and work products do not appear separately in the process models. They are related to each other. These relationships are therefore between variants and variation points as dependences. For example, a product can be an output of a task (the product is produced by the task), so a relationship exists between the task (source) and the product (target). When both the product and the task are variants, the relationships become dependences between variants. Dependences help us to ensure the consistence of the processes created from the process line, and they signify that if an element depends on the inclusion (or exclusion) of another one, then this dependence must be satisfied by means of including (or excluding) another element. It is important to know that dependences are defined in the opposite direction to element relationships because, in order to consider the target, the source must also be considered. Both elements are metamodeled in

Figure 3.

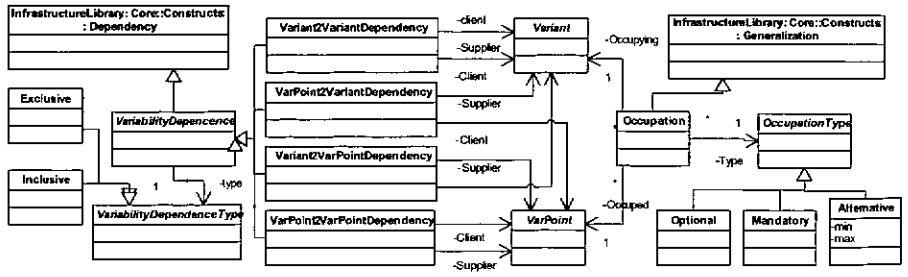


Figure 3: Metamodel of *Variability* Dependences and *Occupation* Relationship

3.1 Process Line Example

The aforementioned elements can be used to define a software process line with which to tailor proc-

esses [9]. We have defined a process line of the *Assessment and Improvement Software Processes* process. This process includes two activities, Assessment and Improvement, and we have defined variability in both of these. Both have a *VPWorkProduct* input to introduce variants of work products (*VWorkProduct*), and the Assessment activity includes a sucessor *VPActivity* (variation point of Activity) in which we can place variants of Activities (*VActivity*) (Figure 4-left). These points can be used to introduce new elements to tailor the process we are going to create. Figure 4-right shows that there are several variants related to the core process in order to occupy the variation points.

To create a tailored process, the variation points of the core process must be occupied by the variants from Figure 4-right. As regards the dependences between the variants (variant2variantDepences) Figure 4-right shows that this means that the Used Improvement Model will be produced by the Model Decision task which uses the Result presentation input work product, and the Improvement Manager role. If we wish to consider the Used Improvement Model it is, therefore, also necessary to considered the Model Decision, the Result Presentation and the Improvement Manager. The resulting tailored process is shown in Figure 5.

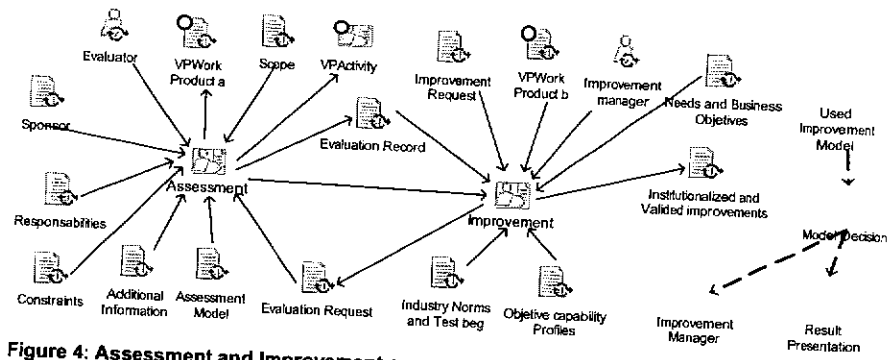


Figure 4: Assessment and Improvement core process (left side) and their variants (right side)

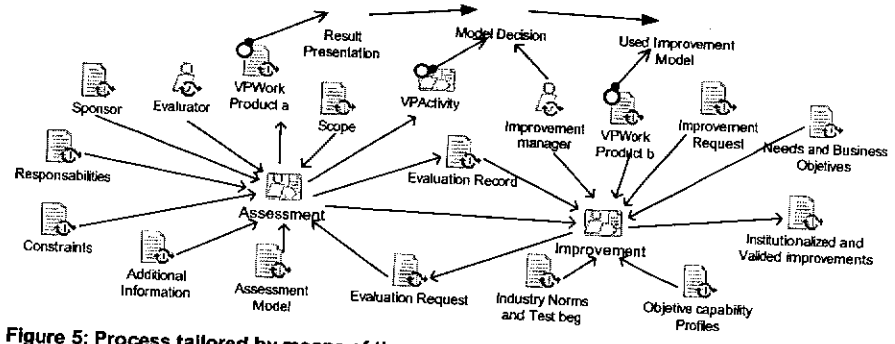


Figure 5: Process tailored by means of the occupation of its variation points with variants

Figure 5 shows a tailored process created from the process line of Figure 4-left, using the variants of Figure 4-right to carry out the "include the User Improvement Model into the Improvement Activity as input work product" variation. It is important to note that since the Improvement Manager role is already part of the core process, it is not included again. Furthermore, this variant cannot be included because the core process has no variation points of type role. This means that if the Model Decision VActivity depends on any other role variant which has not already been defined in the core process, neither the role variant nor the variants depending on it can be included.

4 Enhanced variability mechanisms to built consistent models

Previously we have stated dependences help to ensure the consistence of the tailor processes, and drive how to variants and variation points are related [23]. In the methodological framework we have development; the behaviour of these dependences are vary depending on the elements they related. The behaviours we have described are the following:

- *Variant-to-variant*: these must be used to guarantee that the variants which are going to be inserted do not include inconsistencies between them or the other variants previously inserted into the process when it is going to be tailored.
- *VariationPoint-to-VariationPoint*: these must be used to guarantee that the occupation of variation points are consistently related to the other variation points which are present in the core process and their occupation.
- *Variant-to-VariationPoint*: These allow variability propagation, i.e., when a variant occupies a variation point this may force other variants to be placed as a result of variant-to-variant dependences. However, it may sometimes be of interest to include variability in these other variants. For example, if we insert a task variant, it must be carried out by a role, but it may be of interest to define variability around the role that develops it. These dependences link variation points which are not included in the core process, but are grouped with variants.
- *VariationPoint-to-Variant*: These only signify that the default variant or variants occupy the variation point.

Furthermore, the first two dependences can be defined as either inclusive or exclusive. The first type implies that if the client element is considered, then the supplier element must also be considered. The second type implies that if the client element is considered then the supplier element must not be considered. However, only the last two dependences can be defined as inclusive. A new approach is presented by using a modification of the example from the previous section. The consideration of the aforementioned dependences enables us to carry out a more exhaustive description of the variants, variation points and the occupation relationship between them.

4.1 New Variants

Variants are components of process lines which permit variability. Considering that dependences affect related variants, we can distinguish between two types of variants:

- Root variants. These variants are not involved in any variability dependence as the supplier element, unless the client element of the relationship is a variation point. They may or may not be the client element of any variability dependency.
- Non-root variants: these variants are involved in at least one variability dependency as a supplier element.

The type of a variant is not an inherent characteristic, and may be changed. That is, a variant can be a root variant at one moment (since no variants depends on it), and another variant may later be defined which depends on the first. The first variant then becomes a non-root variant. Therefore, if a variant is deleted (because it will become obsolete) other variants can be converted from non-root variants into root variants.

Specific notations for each type of variant have been created in order to distinguish between them in graphic diagrams. Variants are represented by means of a "V" placed in the top left-hand corner of the SPEM element which is used to create the variant. Therefore, root variants use a black "V", and non-root variants use a white "V". Figure 6 shows this notation when applied to activity variants.

These icons and the dependences described in Section 4, are used to draw some related variants in Figure 7, which shows the difference between the root variant and the others (non-root).

Figure 6: Icons to represent root and non-root variants

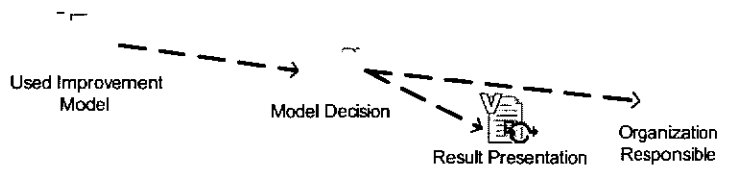


Figure 7: Set of related variants using the new proposed notation and its dependences

Furthermore, most of the variants and variation points in a process line are interrelated by means dependences. This means that a graph is created over the process line. This graph may be unconnected or cyclical, according to the dependences defined. We therefore know that by starting from each root variant it is possible to define a tree of variants which contains the variants the root type depends on by following dependences.

The definition of root and non-root variants is permitted by using the *VariantType* abstract class and two concrete classes associated with the variant class to model the type of variant. Figure 8 shows variant types related to the Variant abstract class, and how concrete root and non-root variants are defined by using the example of activity variants (*VActivity*).

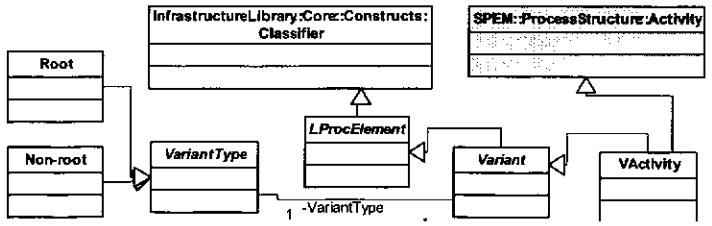


Figure 8: Metamodel of variants including the specification of type.

4.2 Variation points

Variation points are empty points that the process line designer places in the core process to be occupied by different variants, which permits variability. However, processes are large, complex structures; they include several elements and the presence of dependences signifies that the realization of a single variation may imply several variants. Core processes therefore need a greater capability to be varied and to create consistent processes than the explicitly modelled variation points offer.

Two types of variation points can thus be distinguished:

- Explicit variation points. These are the points which are explicitly placed in the core process by the process line designer.
- Implicit variation points. These points appear in the core process exclusively as a result of the definition of the process itself. They are not explicitly defined by the process line engineer.

An example of an implicit variation point might be the set of roles working in a process. This point may be occupied by a role variant. The occupation rules in implicit variation points which ensure consistence in a generated process are shown in Sect. 4.3. Furthermore, a variant may depend on a varia-

tion point, as is explained in Sect. 4. However these variation points must be explicit variation points.

Since implicit variation points are not explicitly included in the process model, they may sometimes not appear in the model. But when they are occupied they must be included in the model. Some new icons have been defined to represent both implicit and explicit variation points with the aim of making diagrams more readable. Figure 9 shows an empty circle to represent explicit variation points and a dotted circle which represents an element that is an implicit variation point. Both implicit and explicit variation points are also shown, using these circles over the original icon.

Figure 9: Circles to represent explicit and implicit variation points, and their application to activities

To allow us to distinguish between both types of variation points, Figure 10 shows the metamodel which portrays variation points as an abstract class associated with a *VariationPointType* class. This class can be defined as an Explicit or Implicit class, to represent the type of variation point. It can be used to define specific variation points, as is shown in the variation points activity (*VPActivity*).

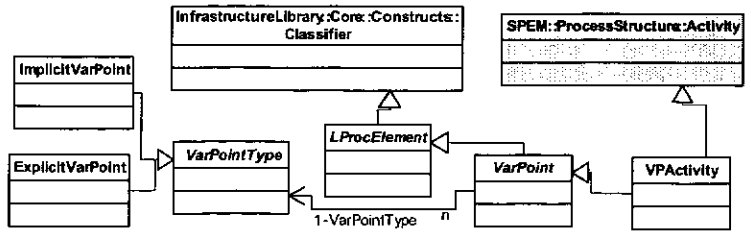


Figure 10: Metamodel of variation points with the specification of their type

4.3 Obtaining tailored processes from software process lines

Obtaining tailored process from a process line involves placing variants in variation points making use of variability. In Section 4.1 root and non-root variants have been defined, along with the implicit and explicit variation points defined in Section 4.2.

Moreover, to achieve correct tailoring it is important know why the variation is carried out. That is, what is the objective of carrying out that variation in the process. The objective is achieved by means of variations and is satisfied with the variants that the process engineer selects to vary the process. That is, from all the variants in the graph, the objective variants are those selected by the process engineer to vary the process. They themselves control the manner in which they and other variants are inserted into variation points. According the variants of Figure 7, we know that the Used Improvement Models root variant is also an objective variant, because it satisfies the "include the User Improvement Model in the Improvement Activity as input work product" objective variation in the core process of Figure 4-left. Other variants are part of the tree created from the objective variant because of their dependences.

Variants can be seen as root or non-root owing to the configuration and dependences of the process line, which implies the type can be seen as a state (Figure 8). However, variants are objective at the moment at which they are on the point of becoming part of the new tailored process through the occupation of a variation point. Objective variants can therefore be distinguished through the occupation relationship which links variants to variation points. The Occupation class in

Figure 3 thus adds the new *isObjective:boolean* attribute in order to distinguish the variants that satisfy

objective variations.

As occurs with root variants, trees can be created by starting from objective variants and following dependences. These trees are composed of root and non-root variants. Used Improvement Models is the objective variant of the variants shown in Figure 7, and the remaining variants are therefore part of the tree it has created.

Furthermore, root variants and objective variants are related. Root variants may be included in a tree of objective variants for two reasons. The first is when the root variant is selected as an objective variant, as occurs in Figure 7. The second is when it is pre-selected as a default variant of an explicit variation point (and related because of a *VariationPoint-to-Variant* dependence). Non-root variants are included in this tree because the root variant depends on it. However, the process engineer may also select a non-root variant as objective variant. In this case, it is considered to be a root variant, its variability dependences as a supplier element are not considered and it defines its own tree.

When objective variation trees are going to be inserted in the core process the occupation of variants in variation points is limited owing to consistence. Variants include the *canOccupy (v:VariationPoint)* procedure shown in Figure 2. This procedure, which is defined for explicit variation points, ensures that variants are placed in suitable variation points because of consistence. As objective variants are selected to vary the process, they must satisfy the *canOccupy* procedure with the explicit variation points they are going to occupy. As regards root variants, if they are an objective variant they must satisfy the *canOccupy* procedure with an explicit variation point. If they are not an objective variant they are included in the tree because they are the default variant of an explicit variation point. Both reasons signify that the root variant must be placed in an explicit variation point and this point will be suitable for the variant. In the case of non-root variants acting as objective variants, they must occupy an explicit variation point as is explained with root variants.

However if non-root variants are not the objective of a variation they are considered in the core process owing to a root variant. There is therefore consistence between both variants as a result of dependences, and there may of course be consistence between the root variant and the explicit variation point of the process (Figure 11). The consistence of non-root variants is thus ensured through the root variant means that it is unnecessary to check the consistency of the points they are going to occupy, as can of course occupy implicit variation points, or explicit variation points. Table 1 summarizes variants that can occupy different variation points.

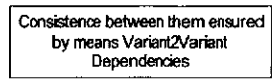


Figure 11: Ensuring consistence between non-root variants and implicit variation points by means of root variants

As we have seen, inserting the variant tree of Figure 7 into the core process of Figure 4-left is not possible because it does not have a role variation point, as was explained in Section 3.1. However, by using the proposed approach, this situation does not occur, since the role in this tree of variants is a non-root role variant and is not the objective variant. It can therefore occupy any of the implicit role variation points in the process, according to Table 1. The set of variants can be thus considered in the core process and it is possible to carry out the proposed variation, as is shown in Figure 11

To sum up, non-root variants are able to occupy implicit variation points. This makes it easy to ensure consistence in tailored process models, because it is possible to place the variants that are not the objective of a variation in generic places of core processes (named implicit variation points). Variants may thus be improved, or can be reused from similar process lines, and will be suitable for use.

Table 1: Occupation rules according to variants and variation point types

Type of variant	Type of variation point it may occupy
Root variant	Explicit variation points
Non-root variant but objective variant	Explicit variation points
Other non-root variants	Either explicit and implicit variation points

5 Conclusions

This work shows that it is necessary to tailor processes to make them usable. A variability mechanism based on variants and variation points additionally makes tailoring possible. These elements are therefore added to SPEM in order to enable process lines and tailored processes to be created. A classification of both variants and variation points, have been presented according to the dependences that are defined between them and their behaviour. Furthermore, we have analyzed how they are used to configure tailored processes and thus how their type affects the way in which they may interact with each other, showing an example of their application.

Overall, this approach facilitates the user's capacity to personalize his/her own processes according to his/her necessities. Tailoring is simplified to the decision of which variants will satisfy the objective variations, and the selection of the variation points they are going to occupy. Moreover, process consistence is automatically ensured in a (from the user's point of view) transparent manner by controlling the behaviour of the elements and managing related variants in the correct way.

This approach is also applicable to process modelling tools to allow them to model process lines, and enables processes to be automatically personalized. What is more important, since variants are atomic elements they can, if necessary, be readjusted during the process execution, and their use can be studied to improve the way in which they are used to improve processes, or to create new variants according to new variability necessities.

Furthermore, considering that variants are organization assets, they can be use to tailor different core process. Other variants can be created in order to better tailor processes. Both variants and core processes can be improved in their own way. The use of the presented approach to configure processes signifies that variants can be used to tailor processes throughout their evolution and improvement.

Future work is focused on three directions. The first is to carry out experiments to prove the understandability of this approach, and its application to real process families will later take place. Its application in the COMPETISOFT [24] process model will be carried out to adapt the model to several organizations in various Latin American countries. The second is to include this approach in process modeling tools with the aim of supporting process line management and process tailoring.

Finally, we are working on process institutionalization environment [8] which will include this approach and offers other capabilities such as the flexibilization of process models during their execution to make them similar to real processes. Furthermore, this makes the realization of process mining over variants and core processes possible by using the resulting executed processes to improve the variants, the core processes and thus the way in which processes are tailored.

Acknowledgments. This work is partially supported by the research into Software Process Lines sponsored by Sistemas Técnicos de Loterías del Estado S.A. within the framework of the agreement of the Innovación del Entorno Metodológico de Desarrollo y Mantenimiento de Software, the Program FPU of the Spanish MICINN, and by the ESFINGE (financed by the Spanish MICINN, TIN2006-15175-C05-05) and INGENIO (financed by the JCCM, PAC08-0154-9262) projects.

6 Literature

1. Clements, P. and L. Northrop, Software Product Lines. Practices and Patterns. 2002, Boston: Addison-Wesley.
2. Osterweil, L., Software Processes Are Software Too, in Proceedings of the 9th International Conference on Software Engineering. 1987, IEEE Computer Press: Monterey, CA. p. 2-13.
3. Rombach, D. Integrated Software Process and Product Lines. in International Software Process Workshop 2005, LNCS 3840. 2005. Teddington, UK: Springer-Verlag.
4. Humphrey, W., Managing the Software Process. 1989: Addison-Wesley.
5. Yoon, I.-C., S.-Y. Min, and D.-H. Bae, Tailoring and Verifying Software Process, in 8th APSEC. 2001. p. 202-209.
6. Ocampo, A., F. Bella, and J. Münch, Software Process Commonality Analysis. Software Process - Improvement and Practice, 2005. 10(3): p. 273-285.
7. Chrissis, M.B., M. Konrad, and S. Shrum, CMMI: guidelines for process integration and product improvement. The SEI series in software engineering. Vol. 1. 2006, Boston: Pearson. 676.
8. Martínez-Ruiz, T., F. García, and M. Piattini. Process Institutionalization using Software Process Lines. in to be published in ICEIS. 2009. Milan.
9. Martínez-Ruiz, T., F. García, and M. Piattini. Towards a SPEM v2.0 Extension to Define Process Lines Variability Mechanisms. in Software Engineering Research, Management & Applications. 2008. Praga: Springer Verlag.
10. Denger, C., F. Elberzhager, and T. Schulz, Customization Approach for Inspection considering Influence and Variation Factors, in BMF Project, F. IESE, Editor. 2008, Fraunhoufer IESE: Kaiserslautern.
11. Ocampo, A. and J. Münch, Software Process Variability: Concepts and Approaches. 2004, Institute for Experimental Software Engineering, Kaiserslautern.
12. Simidchieva, B.I., L.A. Clarke, and L. Osterweil, Representing Process Variation with a Process Family, in ICSP 2007, Q. Wang, D. Pfahl, and D.M. Raffo, Editors. 2007, Springer-Verlag. p. 121-133.
13. Bayer, J., M. Kose, and A. Ocampo, Improving the Development of e-Business Systems by Introducing Process-Based Software Product Lines, in 7th PROFES. 2006, Springer-Verlag: Amsterdam. p. 348-361.
14. Lu, R. and S. Sadiq, A Reference Architecture for Managing Business Process Variants. Proc. 9th International Conference on Enterprise Information Systems (ICEIS2007), 2007.
15. Clauss, M. Generic Modeling using UML extensions for variability. in Workshop on domain specific visual languages. 2001. Tampa Bay, Florida.
16. Webber, D. and H. Gomaa, Modeling Variability in Software Product Lines with the Variation Point Model. Science of Computer Programming, 2004. 53(3): p. 305-331.
17. Schmid, K. and I. John, A customizable approach to full lifecycle variability management. Science of Computer Programming, 2004. 53(3): p. 259-284.
18. Schneiders, A. and M. Weske, Activity Diagram Based Process Family Architectures for Enterprise Application Families, in In Enterprise Interoperability: New Challenges and Approaches, Springer-Verlag, Editor. 2007: London. p. 67-76.
19. Asikainen, T., T. Männistö, and T. Soimimen, Kumbang: A domain ontology for modeling variability in software product families. Advanced Engineering Informatics, 2007. 21(1): p. 23-40.
20. Sinnema, M. and S. Deelstra, Industrial Validation of COVAMOF. Journal of Systems and Software, 2007. 49(1): p. 717-739.
21. Goedicke, M., C. Köllmann, and U. Zdun, Designing runtime variation points in product line architectures: three cases. Science of Computer Programming, 2004. 53(3): p. 353-380.
22. OMG, Software Process Engineering Metamodel Specification. 2007, Object Management Group.

23. Sinnema, M., et al., COVAMOF: A Framework for Modeling Variability in Software Product Families. SPLC 2004, 2004(Springer Verlag): p. 197-213.

24. Oktaba, H., et al., COMPETISOFT: A Improvement Strategy for Small Latin-American Software Organizations, in Software Process Improvement for Small and Medium Enterprises: Techniques and Case Studies, H. Oktaba and M. Piattini, Editors. 2008, Idea Group Inc.

7 Author CVs

Tomás Martínez-Ruiz

Tomás Martínez is a MSc of Computer Science at the University of Castile-La Mancha (UCLM). He is a PdH student at the ALARCOS Research Group of that University. His research interests are software process improvement, software process lines and aspect-oriented software. Contact details are: Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; tomas.martinez@uclm.es.

Félix García

Dr. Félix García is an Associate Professor at the University of Castilla-La Mancha (UCLM). His research interests include business process management, software processes, software measurement and agile methods. He holds MSc and PhD degrees of the UCLM in Computer Science, and is a member of the ALARCOS Research Group of that University, specialized in Information Systems, Databases and Software Engineering. Contact details: Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; felix.garcia@uclm.es.

Mario Piattini

Dr. Mario Piattini has an MSc. and PhD in Computer Science from the Technical University of Madrid and MsC in Psychology from the UNED, and is a CISA, CISM and CGEIT by ISACA (Information System Audit and Control Association). CSQE by ASQ. He is a professor in the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. Author of several books and papers on software engineering, databases and information systems, he leads the ALARCOS research group of the Department of Information Systems and Technologies at the University of Castilla-La Mancha. His research interests are: software process improvement, software metrics, software maintenance and security in information systems. Contact details: Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; mario.piattini@uclm.es

Survey of Practitioners' Attitudes To Software Process Modeling

Tuomas Mäkilä, University of Turku, Finland
Henrik Terävä, Digia Plc., Finland

Abstract

Software process modeling has evolved fast during the past few years. New dedicated modeling standards and process-based tools have been introduced. Emerging trends of the process modeling could bring even more radical changes. These changes have affected the work of common software industry practitioners. Results of a qualitative survey, which was conducted amongst Finnish software practitioners, are presented in this paper. The goal of the survey was to map the attitudes of the practitioners towards the use of software methodologies and software process modeling in their own work. In addition the practitioners provide expert analysis on the emerging modeling trends. The answers of the practitioners are analyzed in this paper and conclusions are given on how the practitioners see the current state and the near future of the software process modeling.

Keywords

Software Process Modeling