Overview of Requirements Engineering Process for Software Product Lines

Snežana Šćepanović, Blažo Popović
University Mediterranean, Faculty for Information Technology, Montenegro

Abstract

Software Product Lines is an important strategy to minimize costs and time-to market, and maximize quality and productivity of the software development. It involves the management of variabilities and commonalities among several applications, which increases its complexity compared to traditional software development. In this context, a Requirements Engineering and management are central tasks, important to reduce the risks involved in a development of product line. System requirements must be properly identified, analysed and reviewed in order to provide adequate solution to manage variabilities and integrating them for making easy the products derivation. In this paper Requirements Engineering process and techniques used in some of the product line practices are reviewed and discussed. Also, Requirements Engineering techniques for traditional single product software development are analysed and their applicability in product line development is assessed.

Keywords: requirements engineering, software product lines, innovation, technology
JEL classification: C88, L86

Introduction

According to Northrop(2008, p.19), 'software product lines (SPL) is set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way'. There are number of benefits that are attractive for organization who strive to adopt SPL approach for product development: improved productivity, increased quality, decreased cost, labour needs and time to market (Northrop, 2008).

However, making a decision that will take an organization to SPL development paradigm is not an easy one. Managers have to estimate cost and benefit for building a software product line, and foresee all risks that are possibly going to arise. Estimation can be done using some of established economic models like Structured Intuitive Model for Product Line Economics (SIMPLE) from Clements et al. (2003). According to SIMPLE model, total cost of development a SPL can be estimated using formula (1):

$$C_{org}(t) + C_{cab}(t) + \sum_{i=1}^{n}\left(C_{unique}(product_i, t) + C_{reuse}(product_i, t)\right) + \\ + \sum_{j=1}^{nbrBenefits} B_{ben_j}(t) \qquad (1)$$

Formula (1) calculate total cost as a sum of four time depended functions: $C_{org}(t)$ - investment needed from organization to adopt product line approach; cost of developing core asset base; $C_{cab}(t)$ - cost of developing core asset base; $C_{unique}(product_i, t)$ -cost of developing unique parts of software product line that are not based on core assets and ($C_{reuse}(product_i, t)$) - cost of software product line development using core asset base (Clements et al., 2003).

It is obvious that development of core asset base during requirements engineering (RE) is most important for organization adopting SPL paradigm. Requirements come from different stakeholders with specific needs and demands for features of the system. In order to develop core asset base, organization need to define adequate RE methodology for SPL that will help managers and system analysts to find common and specific stakeholder requirements.

In this paper RE process and techniques used in some of the product line practices are reviewed and discussed. The paper is organized in four sections. In the first section RE process for SPL is described. Second section presents SPL practices. Third section discusses SPL approach in software development. The Paper finishes with section Conclusion.

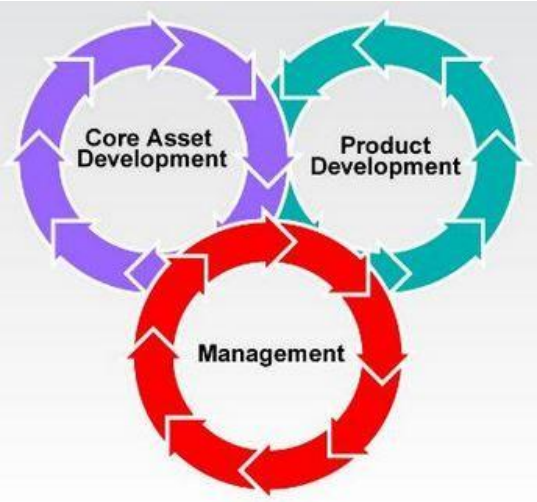
RE process for SPL

There are four main phases in RE process for SPL: *requirements elicitation, requirements analysis, requirements specificatio*n, and *requirements verification*.

In *requirements elicitation* phase, different stakeholders and their requirements for software product line or individual products are identified (Kuloor&Eberlein, 2002). Project team needs to involve domain experts, company managers, end users, power users, sponsors, champions, purchasers and market experts in requirement elicitation. It this phase, it is very important to discover, understand, review and document stakeholder requirements. Each stakeholder presents its own requirements, and project team needs to foresee variation points that can occur in the product line lifetime (SEI – Carnegie Mellon, accessed 25th May 2015).

For successful implementation of SPL approach, development of Core Asset Base is most important goal in the first phase of RE elicitation. For this purpose appropriate RE methodology must be defined at the start of project. In the Core Asset Development activity, which is also called Domain Analysis and Engineering (DA&E), product family requirements and potential family members are identified, analyzed and a reusable domain framework is developed (Kuloor&Eberlein, 2002, p.2). The rotating arrows (Figure 1) indicate that all SPL activities are iterative, linked to each other and can occur in any order.

Figure 1
Software Product Lines Activities



Source: Software Engineering Institute (SEI)

Second phase is *requirement analysis*, which is process of refining and further understanding of gathered stakeholder requirements (SEI – Carnegie Mellon, accessed 25th May 2015). Based on requirement analysis managers and system analysts need to specify: economic benefits for implementation of SPL approach and commonalities and variabilities in stakeholder requirements. These two irreconcilable factors determine the scope, and they model the family of products in product line (Kuloor&Eberlein, 2002). In this phase it is important to define variation points, in order to respond to rapid market changes or the expectations of the customers.

In *requirements specification* phase, all requirements gathered from stakeholders are presented in SPL requirement specification document, used by all involved parties in RE process (SEI – Carnegie Mellon, accessed 25th May 2015). Requirement specifications precisely define and describe product line functional and non-functional (specific) requirements. Final phase of RE process is *requirements verification*. This phase ensures that the right software product is being built and software being developed (or changed) will satisfy stakeholders. Also, requirements verification checks the software requirements specification against stakeholder's goals and requirements.

Requirements engineering techniques for SPL

Software product lines development process differs significantly from single product development process. Successful and comprehensive RE process for SPL needs to combine different RE techniques.

Number of techniques can be used to elicit, gather, document and model stakeholders requirements and their commonalities and variabilities. Since SPL is built for the particular market domain, techniques like stakeholder-view modelling, feature modelling, use-case modelling, change-case modelling and traceability are most appropriate for domain RE (SEI – Carnegie Mellon, accessed 25th May 2015). These techniques are used to explore domain in depth and to help managers to find comprehensive set of SPL requirements and its product specific requirements.

Stakeholder-view modelling technique is used to model important stakeholder requirements (SEI – Carnegie Mellon, accessed 25th May 2015). Model of future system is presented to stakeholders and managers to identify all conflicting requirements and priorities between conflicting requirements.

Feature modelling technique finds commonalities and variation between stakeholder requirements (SEI – Carnegie Mellon, accessed 25th May 2015). After modelling the features using FODA (Feature Oriented Domain Analysis) method proposed by Kang et al. (1990), reference models are created and used to develop products for SPL. Use-case modelling technique is used when variation points are defined in stakeholder requirements. As stated before, variation points are important for managers, to identify economic benefit from SPL. Ways of identify variation points are proposed by Jacobson et al. (1997) and include parameterisation and inheritance.

Change-case modelling is technique used to search and find future changes in the software system (SEI – Carnegie Mellon, accessed 25th May 2015). Designers of the system can include changes in the final design, if they conclude that they will make it more vigorous (SEI – Carnegie Mellon, accessed 25th May 2015). This gives managers the ability to see the design that is going to accommodate future software requirements, and capability to adapt and produce products from SPL that meets market requirements.

Traceability is technique used to map requirements of stakeholders to design and development of product (SEI – Carnegie Mellon, accessed 25th May 2015).

SPL is discipline that is lacking standardized methodologies and processes for gathering comprehensive set of stakeholder requirements. This implies that RE techniques used for single product must be combined to find best solution for SPL RE process. These techniques include interviews, workshops, observations, questionnaires, surveys, system interface analysis, market analysis, and user interface analysis. Kuloor et al. (2002) proposes also soft system methodology (SSM), cooperative requirements capture (CRC) for requirements elicitation phase and structured system analysis (SSA), object-oriented analysis (OOA), joint application design (JAD), quality function deployment (QFD) and participatory design for requirements analysis phase. Software requirement specification and formal methods are proposed by same authors for requirements specification phase. For requirements verification phase, Kuloor et al. (2002) propose prototyping, testing, and requirements reviews.

Software product line practices

In this section, we are presenting requirement process and techniques used in some of the product line practices.

One example is SPL practice is development of Control Channel Toolkit (CCT) by Raytheon Company for the US National Reconnaissance Office (NRO). Software asset base is used to produce ground-based spacecraft command and control systems (Clements et al., 2001). RE methodology for SPL is designed by combining several RE techniques. Process documentation like 'Generalized Requirements Specification' and 'CCT Domain Specification' have shown the strengths of the team that used RE methodology to deal with the huge number of system requirements (Clements et al., 2001). In these documents, commonalities and variations points were clearly identified and modelled combing RE techniques mentioned in the previous section.SPL approach showed reduced development costs, increased quality, and decreased time needed to develop product. From economic point of view, SPL approach for CCT showed significant benefits like reduced costs by 50%,significantly lower number of developers (from 100 to 15), and overall schedule cut by 50% (Clements et al., 2001). Also, total number of SLOC (software lines of code) was 76% less than planned for Spacecraft C2 project who is beneficiary of these systems (Clements et al., 2001).

Similarly, RE process played a major role for development of core asset base called Range Ware, developed by Naval Undersea Warfare Centre (NUWC) for US Department of Defence (Cohen et al., 2002). The core asset development activity involved the creation of the initial core asset base and its use in the development of a small number of products. After the pilot applications of Range Ware, NUWC used the architecture and components to develop new product line systems, improve the core assets, and refine processes such as configuration management for core asset and product development (Cohen et al., 2002). Analysis shows that RE methodology used several RE techniques described in previous section. Implementation of SPL approach showed significant benefits. Cost of building software based on Range Ware core asset base was 50% lower than using traditional approach, development time is reduced and personnel is cut by up to 75%, which means less expenses for the employers(Cohen et al., 2002). Generally, development of core asset base for SPL, enabled company to save 4 million dollars in development and 10 million dollars in maintenance costs, versus non product line approach (Cohen et al., 2002).

Discussion

Success of software product line development largely depends on the RE phase and management. This activities focus on discovering, analysis, documenting and managing system requirements which create core asset base.

SPL cases studies (SEI institute; Brownsword et al., 1996; Clements et al., 2001; Clements&Northrop, 2002; Clements et al., 2005; Cohen et al. 2002) showed that SPL development paradigm has benefits against single software development approach. It can be noted that development of core asset base is most critical for successful implementation of SPL. Also, SPL approach showed reduced development costs, decreased time needed to develop product without affecting quality of product.

Literature review (Jirapanthong, 2011; Kuloor&Eberlein, 2002), has pointed out the software product line development is more complex and demanding than individual software product development, and there are still many difficulties in implementation of this approach. Some difficulties are: necessity of having a basic understanding of the variability consequences during the different development phases of software products; establishing relationships between product members and product line artefacts, and relationships between product members artefacts; poor support for capturing, designing, and representing requirements at the level of product line and the level of specific product and poor support for handling complex relations among product members, for maintaining information about the development process (Jirapanthong, 2011,pp. 41-42.)

Exact methodologies for SPL RE process are still not defied. It is not easy to find right approach to manage large amount of stakeholder requirements in order to develop set of products that have a majority of features in common and vary only in certain specific features. For successful implementation of SPL development paradigm number of software RE techniques needs to be employed.

New approaches and models (Alferez et al., 2008; Huysegoms et al., 2010) are proposed for standardisation the RE process for SPL. Due to the complex nature of product line development, it is essential to have a systematic approach to RE process during software life cycle.

Our review presented in this paper has some limitations. To the extent that we performed a literature review, the potential for incomplete identification of relevant studies and publication are always consideration. In most cases, data collection and analysis were poorly described. Overall, empirical studies are not supported by accurate evidence. A potential risk that we might have missed relevant papers is due to lack of agreed terminology in the SPL field, leading to the possible existence of relevant papers that do not explicitly mention the keywords we specified. To minimize this possibility, the search for potentially relevant studies included a search with multiple databases, and also bibliographic searches of the reviewed articles to identify additional studies. However, we might assume that by combining the list of retrieved papers, we provided a good coverage of publications of the RE practices for SPL.

Conclusion

This paper discussed requirements engineering for software product lines. Various single product requirements engineering techniques were briefly described. Also, requirements engineering practices used in the existing product line approaches are analysed and the applicability of single product requirements engineering techniques in product line development is assessed.

The software engineering literature has pointed out the software product line development is more complex and demanding than individual software product development, and there are still many obstacles in implementation. From the analysis of current practice it is clear that software product line is not suitable for all projects. Organisations need to select techniques that best correspond to their context.

Based on the findings of literature review we suggest that further research into this topic should include empirical assessment of existing strategies, as a means to improve their quality and enable generalizations. This may involve investigating large-scale and industry-side scenarios.

References

1. Alferez, M., Kulesza, U., Souza, A., Santos, J., Moreira, A., Araujo, J., Amaral, V. (2008), "A Model-Driven Approach for Software Product Lines Requirements Engineering", Dept. Informatica, FCT, Universidade Nova de Lisboa, Portugal. The authors are partially supported by EU Grant IST-33710: Aspect-Oriented, Model-Driven Product Line Engineering (AMPLE).
2. Brownsword, L., Clements, P. (1996), "A Case Study in Successful Product Line Development", Technical Report CMU/SEI-96-TR-016, Software Engineering Institute, Carnegie Mellon University.
3. Clements, P., Bergey, J. (2005), "The US Army's Common Avionics Architecture System (CAAS) Product Line: A CAse Study", Technical Report CMU/SEI-2005-TR-019, Software Engineering Institute, Carnegie Mellon University.
4. Clements, P., Cohen, S., Donohue, P., Northrop, L. (2001), "Control Channel Toolkit: A Software Product Line Case Study", Technical Report CMU/SEI-2001-TR-030, Software Engineering Institute, Carnegie Mellon University.
5. Clements, P. C., McGreggor, J. D., Cohen, S. G. (2005), "The Structured Intuitive Model for Product Line Economics (SIMPLE)", Technical Report CMU/SEI-2005-TR-003, Software Engineering Institute, Carnegie Mellon University.
6. Clements, P. C., Northrop, L. M. (2002), "Salion, Inc.: A Software Product Line Case Study", Technical Report CMU/SEI-2002-TR-038, Software Engineering Institute, Carnegie Mellon University.
7. Cohen, S., Dunn, E., Soule, A. (2002), "Successful Product Line Development and Sustainment: A DoD Case Study", Technical Note CMU/SEI-2002-TN-018, Software Engineering Institute, Carnegie Mellon University.
8. Huysegoms, T., Snoeck, M., Dedene, G. (2010), "Building a Requirements Engineering Methodology for Software Product Lines", Faculty of Business and Economics, Management Information Systems Group, Belgium. Work is funded by KBC Global Services NV through the KBC research chair "Developing and Managing Business Services as Shared Assets".
9. Jacobson, I., Griss, M., Jonsson, P. (1997), "Software Reuse Architecture, Process and Organization for Business Success", Addison-Wesley Professional, New York.
10. Jirapanthong, W. (2011), "Experience on Requirements Engineering for Software Product Lines", Journal of Information Science and Technology, Vol. 2 No. 1, pp. 41-49.
11. Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E., Peterson, A.S. (1990), "Feature-Oriented Domain Analysis (FODA) Feasibility Study", Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University.
12. Kuloor, C., Eberlein, A. (2002), "Aspect Oriented Requirements Engineering for Software Product Lines", The University of Calgary, Calgary, Canada.
13. Northrop, L. (2008), "Software Product Line Essentials", Software Engineering Institute, Carnegie Mellon University.
14. Software Engineering Institute (SEI), Carnegie Mellon University (2015), "A Framework for Software Product Line Practice – Version 5.0", , Product Line Practice Initiative, available at: http://www.sei.cmu.edu/productlines/frame_report/index.html (accessed May 25th 2015)

15. Software Engineering Institute (SEI), Carnegie Mellon University (2015), "Product Line Essential Activities", available at: http://www.sei.cmu.edu/productlines/frame_report/PL.essential.act.html (accessed May 25th 2015)

About the authors

Dr Snežana Šćepanović is an Associate Professor at the Faculty for Information Technology, University "Mediterranean", Podgorica, Montenegro. Within different projects at the University level she is also responsible for development of software for e-learning and online study programmes at different levels of study. She is engaged in several national and EU projects related to development of new and alternative methods for lifelong learning using new technologies. Her research interests include: System requirement analysis, Usability and design of GUI, Human computer interaction, E-learning. She participated as project manager and expert in number of EU and national projects. Author can be contacted at **snezana.scepanovic@unimediteran.net**

Spec.Sci Blažo Popović is student on master studies at Faculty of Information Technology, University "Mediterranean" Podgorica. His specialization includes system and network administration of various platforms (Linux, Windows) and software development techniques. He has experience in undergraduate teaching using modern technologies. His research interests include: system requirements analysis, software engineering, embedded software systems, and mission-critical software systems. Author can be contacted at **popovic.blazo@gmail.com**