A Product Line Enhanced Unified Process

Weishan Zhang[1] and Thomas Kunz[2]

[1] School of Software Engineering, Tongji University,
No. 4800 Cao'an Highway, Shanghai, 201804, China
zhangws@mail.tongji.edu.cn
[2] Department of Systems and Computer Engineering, Carleton University,
1125 Colonel By Drive, Ottawa, Canada K1S 5B6
tkunz@sce.carleton.ca

Abstract. The Unified Process facilitates reuse for a single system, but falls short handling multiple similar products. In this paper we present an enhanced Unified Process, called UPEPL, integrating the product line technology in order to alleviate this problem. In UPEPL, the product line related activities are added and could be conducted side by side with other classical UP activities. In this way both the advantages of Unified Process and software product lines could co-exist in UPEPL. We show how to use UPEPL with an industrial mobile device product line in our case study.

1 Introduction

The Unified Process (UP) or Rational Unified Process (RUP) [6] is one of the most popular and complete process models that have been used by developers in recent years. The main characteristics of UP are:

1. Using iterative and incremental development that has a lifecycle consisting of several iterations;
2. Centering around software architecture, which is the highest-level concept of a system in its environment;
3. Embracing change by considering feedbacks from stakeholders and then make corresponding adaptations.

This architecture-centric approach is facilitating reuse for a single system development. Although it is claimed in RUP that "it also allows reuse on a larger scale: the reuse of the architecture itself in the context of a line of products that addresses different functionality in a common domain", in reality, it is very difficult to achieve this goal without the related supporting technology. There are no mechanisms in RUP to handle the technical issues for a software product line [1], for example variability management.

The current object-oriented technology and component-based development (e.g., with .NET™ or J2EE™), recommended in the RUP practice, provide many useful reuse mechanisms, but in many instances fail to achieve the desired reusability and maintainability. The main reasons come from the intrinsic problems of current programming languages and development methodologies [11].

Q. Wang et al. (Eds.): SPW/ProSim 2006, LNCS 3966, pp. 142 – 149, 2006.

In this paper, we present an enhanced Unified Process called UPEPL (Unified Process Enhanced with Product Line) that incorporates the product line technologies implemented with XVCL (XML based Variant Configuration Language) [11]. The architecture of the underlying system(s) is implemented as a hierarchy of meta-components, which is called an x-framework in XVCL jargon. UPEPL was demonstrated with an industrial mobile device project in which we achieved good reusability, development and maintenance gains.

The rest of the paper is structured as follows: Section 2 presents the UPEPL process in which product line related activities are added and could be conducted side by side with other classical UP activities; then we demonstrate this process with the creation of a mobile game product line. In Section 4, we discuss our case study using the UPEPL process. The related work and concluding remarks end the paper.

2 Unified Process Enhanced with Product Line Technology

In UPEPL, the integration of the product line into the Unified Process may start from the end of the first iteration, or after the release of some products in the product family just like the typical process of product line engineering. Here we show the related activities in UPEPL starting from the early start of the inception phase.

There are four phases in the Unified Process, namely Inception, Elaboration, Construction, and Transition. The inception phase 'focused on ensuring that the project is both worth doing and possible to do', including the business case and the scope of the system. In the elaboration phase, the architecture is created and validated, which lays the foundation for the following activities. The construction phase focuses on the development of the system according to the baselined architecture. The product is delivered to the end user in the transition phase. As the transition phase is relatively simple, we will elaborate other three phases that incorporate product line related activities.

The activities shown in the following figures will follow the style defined in UP, with slight modifications where necessary, and activities and artifacts related to product line are shown with italic fonts. To make the figures clear and concise, the supporting activities, for example the change control, are not shown.

2.1 Inception

As shown in Fig. 1, besides the activities in a normal UP, the inception phase in UPEPL involves additional activities for product line visioning, for example, product line scoping to explore the degree of the commonality and variability, and conducting the initial domain analysis, in order to decide whether the product line is feasible or not.

The main artifacts produced in the inception phase are outlined software requirements, proof-of-concept software architecture, and initial domain feature model. Software requirements are organized in a use-case specification document, and a supplementary requirement document. As the non-functional requirements may appear both in the use-case specification and the supplementary requirements, and there are many similarities for a use case in different product line members, we are using the requirement x-framework to remove the redundancies as proposed in [8,9]. This will keep different documents in consistency.

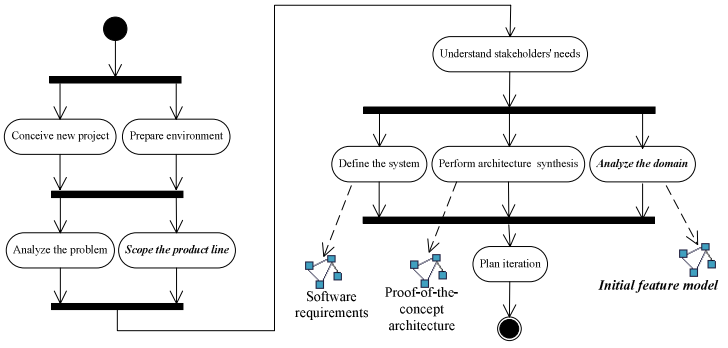

Fig. 1. Activities in inception phase

2.2 Elaboration

In the elaboration phase (Fig. 2), additional activities in UPEPL are related to the feature model refinement, product line architecture development, etc. The proof-of-concept architecture is refined to satisfy the requirements of the first product line member, and this serves as the foundation to develop the first-cut product line architecture (PLA).

More variants and commonalities could be identified during the process of the domain analysis in elaboration phase. This leads to the refinement of the product line requirement (represented as an x-framework), feature model, and product line architecture. The incorporation of the variants into various assets was discussed in previous work [4, 5].

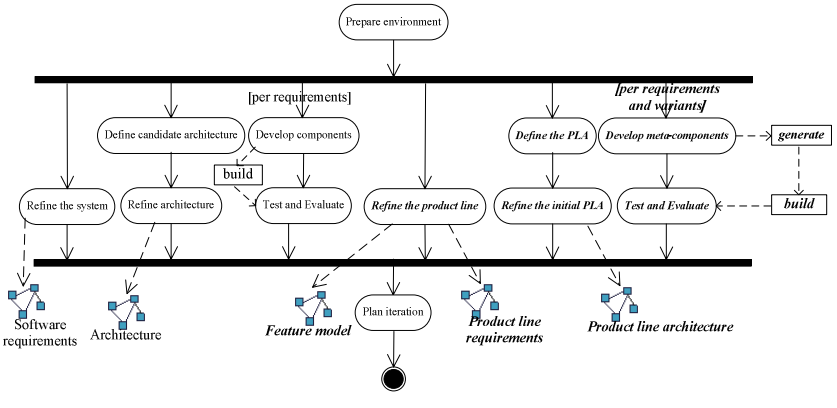

Fig. 2. Activities in elaboration phase

Meta-components for all kinds of assets (including requirements, models and code) are developed incrementally. The first set of meta-components may stem from a typical system and only address its own variants. As the developments proceeds, more variants

will be added to make the meta-components more adaptable. And also the related meta-architecture becomes more evolvable as more domain variants are resolved.

Meta-components are used to generate specific components according to the specification for a product line member. Therefore an additional 'generate' process is added before the 'build' starts. The build activity may not be required if the generating process is not for code components, but for requirements, models and other documentary assets.

2.3 Construction

More meta-components are developed in the construction phase. After the product line is ready, the development of a new product line member may involve the selection of the meta-components from the meta-component repository. The selection process starts from the examination of the feature model in order to select the appropriate variants, and then adapting them by writing specification meta-components and modify related meta-components where necessary.

Unit testing and integration testing are also performed in this phase. The product is evaluated against the acceptance criteria in order to make a smooth transition to the end user. Activities in the construction phase are shown in Fig. 3.

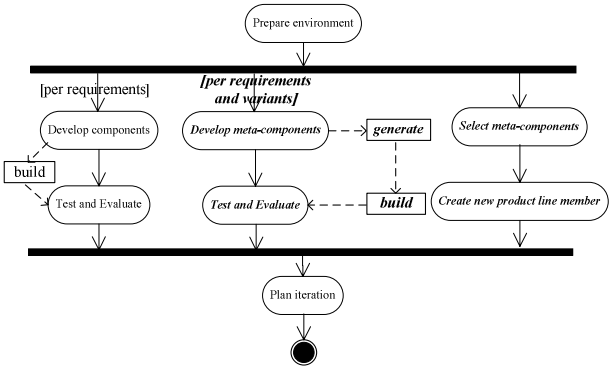

Fig. 3. Activities in construction phase

3 Case Study with a Mobile RPG Product Line

Mobile gaming is becoming increasingly popular. With a Role-Playing Game (RPG), the players take the roles of fictional characters and participate in the interactive story. The player's decision-making drives the story forward and the outcome varies depending on the players' actions.

For starting we will consider the ***Climb*** game (Figure 4) where the hero jumps up and down the floor (a bar in the following screen), in order to avoid falling down to the bottom of the mountain. Time elapsed and remaining is displayed with a thin bar on the top of screen.

Fig. 4. Climb game screen shot

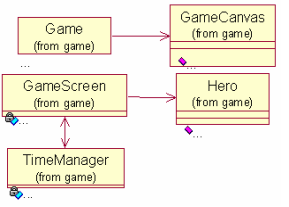

Fig. 5. Common concepts in mobile RPG domain

3.1 Inception

First we will consider four RPGs including Climb we just introduced; ***Kongfu*** where a young man learns kongfu skills from his 'master'; ***Feeding*** where the hero tries to pick up as much food as possible; In ***Hunt***, the hero shoots animals and monsters with arrows. All games are implemented with MIDP2.0 in J2ME platform.

When we look at all these RPGs, we do find some commonalities and variabilities among them. For the commonalities, we can find that there are always heroes in the game scenario, scores are increased or decreased, etc. It is very natural to consider these RPGs as a game product line. A mobile RPG product line should bring promised advantages over the classical development.

The initial examination into the code verified this as there are many similar code patterns inside the games. We analyzed this with our own code clone searching tool called JCloneMiner.

To save space, we do not show the initial feature diagram and other related documents here.

3.2 Elaboration

In the first iteration of the elaboration phase, we further analyzed the mobile RPG domain. The common concepts (implementation with MIDP) are illustrated with a UML class diagram (Fig. 5).

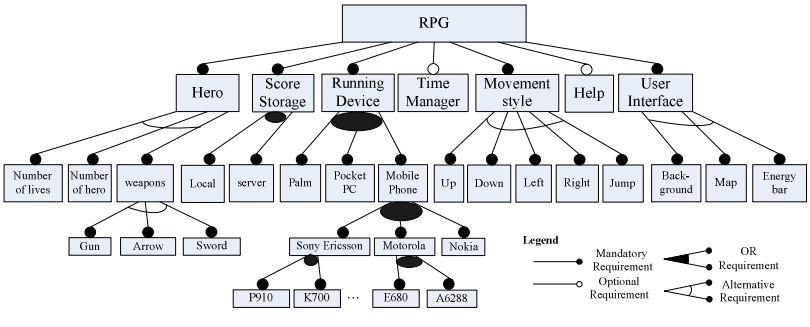

Fig. 6. Feature model for the mobile RPG product line

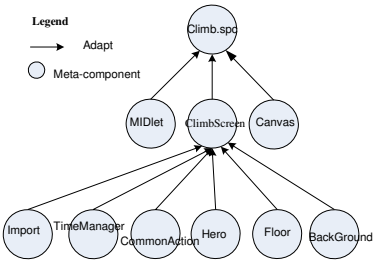

Fig. 7. the first-cut RPG PLA

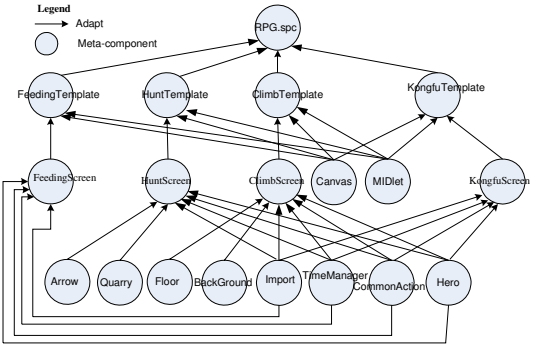

Fig. 8. the final RPG PLA

The feature model is shown in Figure 6. Please note that the feature model may be refined as the iteration goes.

The first-cut RPG product line architecture (RPG PLA) was created by identifying and developing meta-components starting from the Climb game, as shown in Figure 7.

In the second iteration, more meta-components were developed. The first-cut PLA was refined to incorporate more domain commonalities and variants. During the creation of the meta-components and the refinement of the PLA, some of the optimizations were found and incorporated in the related meta-components, which will benefit all components generated from these meta-components. This was discussed in more detail in [10]. The final PLA is shown in Figure 8.

3.3 Construction

Since we have created the mobile RPG product line architecture, we can reuse it in the construction phase to develop a product line member. For example, assume we want to develop a game called ***Dig gem*** (Figure 9). The hero digs around the map to look for various kinds of gems. Different scores for different gems will be added to the total shown on the top. There may be traps and bombs which will consume the energy of the hero. Time elapsed and remaining is displayed with a bar on the top of screen.

Some of the meta-components, such as Hero, TimeManager, etc. could be reused. But other components, for example Cloud, PopUpMenu must be developed and added to the meta-component repository for future reuse.

Fig. 9. screen shot of the Gig gem game

4 Discussion of the Case Study

We first use the typical Unified Process (which is also part of UPEPL) to develop the four games, then we apply the UPEPL to incrementally build the mobile RPG product line. This process is summarized in the following table. The reduced lines-of-code (LOC) count is shown.

Table 1. The process of applying UPEPL to build the RPG product line

	Original LOC	Meta-components LOC	Reduced LOC	Reduced Percentage
climb	941			
feeding	463			
subtotal	1404	1154	250	17.8%
kongfu	720			
subtotal	2124	1579	545	25.7%
hunt	1286			
Total	3410	2547	863	25.3%

From the above table we can see that for a new product line member that is very similar to some of the existing members, the development efforts may decrease steadily. But for a member who has more differences, the reuse ratio may decrease a bit. In the case of the Kongfu game, as there are two heroes with different roles, the Hero meta-component was adapted two times in order to generate specific code components for the two heroes respectively. Therefore the reduced code percentage increases greatly.

In this process, the design and implementation of the mobile games could be unified, which is very important for software design and maintenance. The configurability from XVCL will make the changes to the related components in a consistent way.

6 Related Work

Gomaa presented PLUS method in his work [2]. PLUS is a design method for software product lines that describes how to conduct requirements modeling, analysis modeling, and design modeling for software product lines in UML. He also discussed the integration of PLUS with Unified Process. In essence, his method uses the same methodology as UPEPL. In UPEPL, we use a specific product line technology implemented with XVCL.

Massoni proposed RUPim [7] in order to support progressive and separate implementation of persistence, distribution, and concurrence control. This will reduce the impact of requirement changes, and simplify testing and debugging. UPEPL is aimed to improve the reusability and productivity hence, to inject the strong points of product line technology into the traditional UP.

Other extensions include RUPSec [3] dedicated to security system, where threats and security requirements can be captured and modeled by adding new Roles, Activities and Artifacts. If needed, such extensions could be added to UPEPL too to address specific application domain issues.

6 Conclusions and Future Work

We have proposed a product line enhanced Unified Process called UPEPL. The typical activities in the Unified Process could proceed side by side with product line related activities. UPEPL is demonstrated with a mobile RPG product line, in which four games were considered to build the RPG PLA. It shows that UPEPL is an efficient approach where both the advantages of Unified Process and software product lines can co-exist.

In the future, further applications of UPEPL will be conducted with other domains, such as the CRM systems. We are also considering developing an integrated development workbench, in which meta-component mining, smarting editing and debugging are all included. The UML modeling part are to use Rational Rose, where a plug-in should be developed to link them together.

Acknowledgements

This research is sponsored by "Excellent Young Teacher Funds of Tongji University". Thanks to Liu Wei and other participating develops from Meitong Co. Ltd.

References

1. Clements, P. & Northrop, L. "Software Product Lines: Practices and Patterns". Addison-Wesley, 2001.
2. Gomaa, Hassan.Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures. Addison Wesley, 2004
3. Jaferian, P. Elahi, G., Reza, M., Shirazi, A., Sadeghian, B. RUPSec: Extending Business Modeling and Requirements Disciplines of RUP for Developing Secure Systems. Proc. of EUROMICRO-SEAA'05. Porto, Portugal, Aug. 2005
4. Jarzabek, S., Zhang, H.: XML-Based Method and Tool for Handling Variant Requirements in Domain Models. RE 2001: 166-173
5. Jarzabek, S., Wai Chun Ong and Zhang, H. Handling Variant Requirements in Domain Modeling. SEKE 2001: 61-68
6. Kruchten, Philippe. The Rational Unified Process, An Introduction, Second Edition. Addison Wesley Longman, 2000
7. Massoni, TL. A RUP-Based Software Process Supporting Progressive Implementation, in book 'UML and the Unified Process', Idea Group Publishing, 2003
8. Zhang W., et al. Software evolution with XVCL. A chapter for the book "Software Evolution with UML and XML", Idea Group Publishing, Dec. 2004
9. Zhang W. Architecturally Reconfigurable Development of Mobile Games. Proc. of the ICESS2005, Xi'an, China, IEEE CS, December 2005, pp 66-72
10. Zhang, W., Jarzabek, S. Reuse without Compromising Performance. Proc. of SPLC2005, Rennes, France, September 2005, Springer LNCS3714, pp. 57-69
11. XVCL homepage. http://fxvcl.sourceforge.net