

Requirements Engineering for Software Product Lines

Chethana Kuloor Armin Eberlein,

The University of Calgary,

2500 University Drive, N.W, Calgary, Alberta, Canada – T2N 1N4

Tel: (403) 220-5002, Fax: (403) 282-6855

Email: chethana-kuloor@shaw.ca, eberlein@enl.ucalgary.ca.

Abstract

The cost of fixing a requirements defect later in the development stage is much higher than the cost of identifying and fixing it in the early stages of development. In order to do this the system requirements must be properly identified, analyzed and reviewed early in the development process. Requirements engineering is such a process that focuses on discovering, analyzing, documenting and managing system requirements. Several processes and techniques have been developed to assist requirements engineering activities. A product line is a set or group of products that have a majority of features in common and vary only in certain specific features. Developing a group of products that have a majority of features in common supports a great deal of reuse in all the phases of system development. Due to the complexity and extensive nature of product line development, requirements engineering is much more important for a product line practice. In this paper an attempt is made to review requirements engineering techniques used in existing software product line practices. Also the applicability of several requirements engineering techniques to software product line development is assessed.

Keywords:

Requirements Engineering, Software Product Lines, Software Product Families, Domain Analysis and Engineering.

1. Introduction

The use of computers and software products has enormously increased in the past several years. There is an increase in the demand as well in the complexity of the software needed. In order to obtain high-quality products along with higher productivity, it is required to carefully analyze, model, specify and manage system requirements. This would not only simplify system design and implementation but also reduce the number of defects that are identified later in the implementation stage. Requirements engineering is introduced to address such issues early in the development process. A well-established requirement engineering process ensures that system requirements are properly elicited, analyzed, documented, verified and managed.

Several other attempts have been made to increase the productivity and quality of software products. A very promising approach is software reuse. Software product line practice is one approach to software reuse, which focuses on developing a family of systems. The main goal is to develop a framework that represents the family, which is then customized to develop individual products. A product family is a collection of similar products that share most of the features. Hence there are numerous requirements that are common across the family but others are unique to individual products. There are several customers and stakeholders involved. Therefore a well-established requirements engineering process is essential for any product line practice.

In this paper, requirements engineering practices used in the existing product line approaches are analyzed and the applicability of several existing requirements engineering techniques to product line development is assessed. This paper is structured into two major sections: Software Product Line Practices and Requirements Engineering Techniques. In the section on Software Product Line Practices various product line practices are listed and requirements engineering activities used in each of them are reviewed. In the second section various requirements engineering activities are briefly described and their applicability to software product line approaches is assessed.

2. Software Product Line Practices

The process of developing a set of products that share a majority of features is called Software Product Line development practice. A typical product line practice has two important activities: The Core Asset Development and the Product Development [1]. 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. In the Product Development activity the domain framework is instantiated to develop individual applications. However, in the existing product line practices these activities are carried out in several different stages. For example Synthesis and FAST have only two stages: Domain Engineering and Application Engineering [2], [3]. In several other approaches such as FODA, PuLSE, Sherlock and Odyssey-DE they are carried out in more than two stages [4], [5], [6].

In this paper, we are interested in the requirements engineering processes and practices used in product line practices. In product line development requirements engineering activities are carried out in the early stages of DA&E. Requirements engineering processes and techniques are used to identify and characterize product line requirements and potential member products. They also involve activities such as product line requirements modeling, specification, verification and management. Requirements process and techniques used in some of the product line practices are reviewed below.

2.1. Synthesis

Synthesis provides a framework for product line development that is customizable to different organizational contexts [2]. Organizations can adopt Synthesis fully or partially based on their reuse level. In Synthesis product line development activities are grouped into two stages: Domain Engineering and Application Engineering. Figure 1 shows the activities involved in the domain engineering stage. Initial activities such as Domain Definition, Domain Specification and Domain Verification are responsible for identifying and characterizing product line requirements and the potential family members. In the Application Engineering stage individual applications are developed.

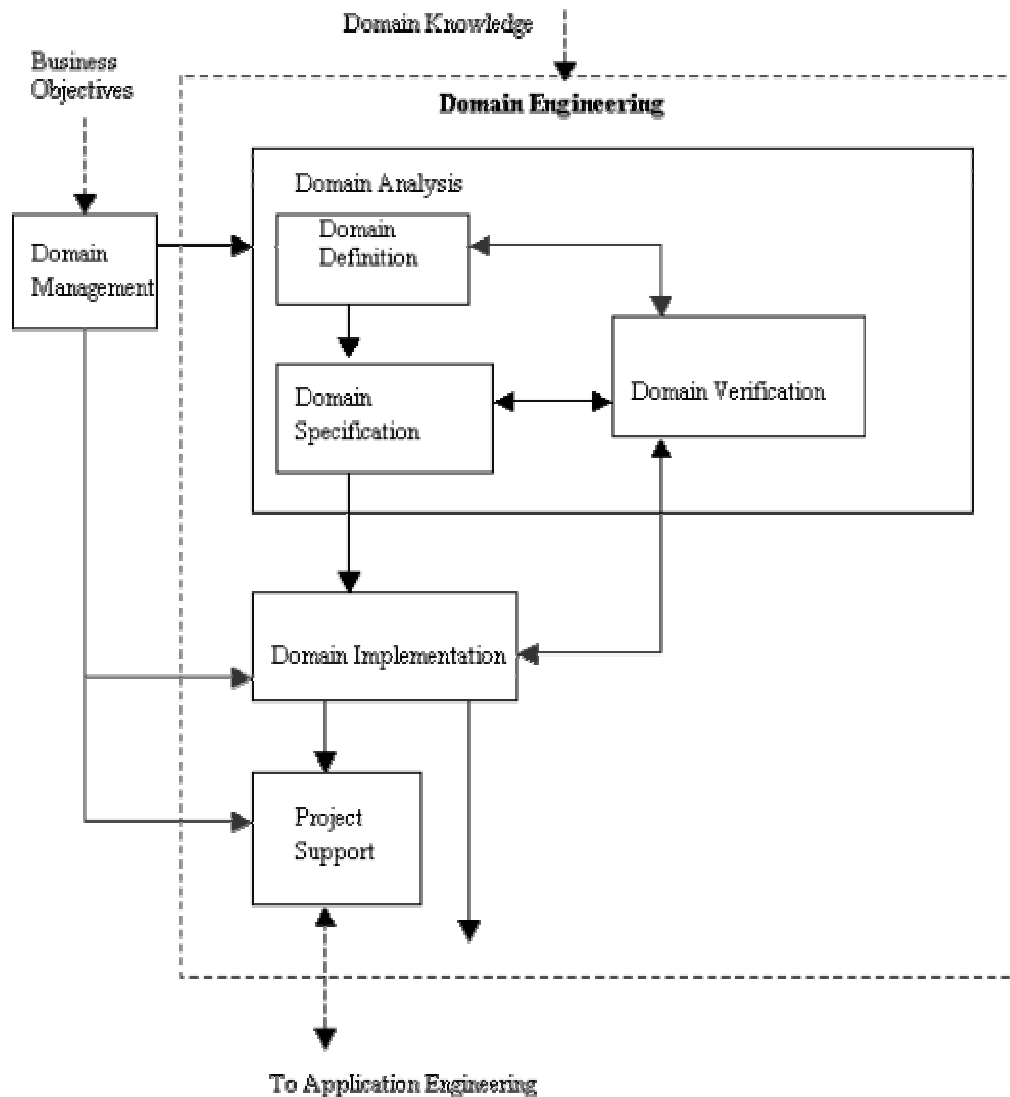


Figure 1. Domain engineering activities in Synthesis [2].

Synthesis is a framework that provides guidelines for product line development. It also provides a framework for product line requirements engineering. It suggests having several distinctive stages such as Domain Definition, Domain Specification, Domain Verification and Domain Management that can correspond to

requirements capturing, analysis, specification and management. However, it does not specify or recommend any specific techniques for carrying out most of the above activities. It suggests having a specification template for representing domain work products. It also provides guidelines for managing various domain activities. However, it provides very little information about representing and navigating the dependencies between requirements and other domain entities.

2.2. Family Oriented Abstraction, Specification and Translation (FAST)

FAST is a systematic process for developing a set of systems, which share a majority of common features [3]. FAST has two major sub-processes: Domain Engineering and Application Engineering. In the Domain Engineering sub-process, requirements for the product family are defined and the reusable assets required for building the family members are developed. In the Application Engineering sub-process, the individual members are developed using the reusable assets according to the specification. In FAST common and variable requirements for the family are defined and analyzed using a process called “Commonality Analysis” [4]. This process is a structured group discussion between several domain experts. The main purpose is to elicit product family requirements and to document them. These requirements are then specified using a domain specification language so that a translator can automatically translate these domain specifications into individual products.

FAST is well suited to develop applications quickly. It includes several activities that are equivalent to requirements engineering activities. The requirements elicitation and analysis process in FAST is called Commonality Analysis where a group of domain experts gather and analyze family requirements. Commonality Analysis is a good process for gathering information about the domain since it involves a structured discussion between domain experts and requirements engineers. However, it does not recommend any specific technique for capturing, analyzing and characterizing the family. FAST uses certain domain specification languages to represent domain work products. Requirements can also be represented using the same specification language. Since FAST uses a translator, which automatically generates individual applications, there will be a drastic reduction in the time required for development and testing. However, since requirements are specified using custom domain specification languages, other organizations may have difficulties in adopting them. For system specifications to be translated automatically, they must be precisely defined and specified. If there is a change or error in the requirements, specifications have to be modified and the translation needs to be altered. Tool support is provided for several activities. Very little attention is given to requirements management and traceability. If the family requirements are properly identified and characterized through the Commonality Analysis process, FAST can develop applications very quickly.

2.3. Feature Oriented Domain Analysis (FODA)

Feature Oriented Domain Analysis (FODA) is a domain analysis and engineering technique, which focuses on developing reusable assets [5]. FODA has three phases: Context Analysis, Domain Modeling and Architecture Modeling. In the Context Analysis phase, information required for various activities are gathered from various sources. In the Domain Modeling phase, product line requirements are analyzed using a set of domain models. Common and variable requirements are identified using a technique called Feature Modeling [4]. Feature models consist of feature diagrams that represent features in a hierarchical structure. These requirements are further analyzed using several structured system analysis techniques such as data-flow diagrams, entity-relationship diagrams and functional diagrams. The end users and domain experts review these domain models. In the Architecture Modeling phase, the domain models are used create an architecture model. The architecture model can be instantiated to develop individual applications.

FODA is a domain analysis and engineering approach that makes use of structured system analysis techniques such as data flow diagrams and entity relationship diagrams. FODA has some requirements engineering activities such as requirements capturing and requirements analysis. The requirement capturing process in FODA is called “Context Analysis” where information about the target and related domains are gathered from various sources. Requirements are analyzed and modeled in the Domain Modeling stage. Feature Modeling is a very good technique that can be part of the requirements analysis stage to identify common and variable

requirements. When the features are being represented in a tree like structure, it is easier to identify the optional and alternative features. In addition to this, features represented in a hierarchical structure would be easier to traverse and understand. However, structured system analysis techniques are not very good techniques for reuse since they do not encourage component-based development. FODA does not have any specific process or technique for requirements specification, verification and management.

2.4. Product Line Software Engineering (PuLSE)

Product Line Software Engineering (PuLSE) is a customizable product line practice [6]. PuLSE can be applied to a variety of enterprise contexts through customizability of components, incremental introduction capability and a maturity scale for structured evolution. A PuLSE process has three basic elements: Deployment Phases, Technical Components and Support Components. Deployment Phases are logical steps in the PuLSE practice that describe the activities needed to define and develop a family of products. These activities are categorized under three stages namely PuLSE Initialization, PuLSE Infrastructure Construction and PuLSE Infrastructure Usage. In the PuLSE Initialization stage a PuLSE process that is suitable for the current domain and organizational context is defined. In the PuLSE Infrastructure Construction stage, the product line characteristics are defined, modeled and a reusable architecture is developed. In the PuLSE Infrastructure Usage, the product line architecture is instantiated to develop individual applications. PuLSE technical components include the technical expertise required to carry out various PuLSE activities. PuLSE support components provide the guidelines required to solve any non-technical issue such as organization issue, project entry points and process evolution. Kobra is an object-oriented customization of PuLSE [7]. However, Kobra focuses more on the design and implementation of a domain framework.

Similar to Synthesis, PuLSE provides a framework for product line development. Although it proposes several distinctive stages for product line development, it does not have a systematic process for requirements engineering. It allows the organizations to decide on the tools and techniques required for each activity. However, it uses a technique called Product Map for product line scoping and characterization. A Product Map is a technique that maps the product line features with member products in the form of a matrix. Using the subjective evaluation on the importance and market benefit, certain characteristic and benefit functions are calculated. These functions are then evaluated to choose the product line requirements and member products. Product Maps are a good technique that can be used to identify product line features and potential members. PuLSE provides very little information about requirements specification, verification and traceability.

2.5. Sherlock

Sherlock is an object-oriented domain analysis and engineering practice [8]. In Sherlock a domain framework is developed using five basic steps. They are Domain Definition, Domain Characterization, Domain Scoping, Domain Modeling and Domain Framework Development. During Domain Definition, Domain Characterization and Domain Scoping information from various sources are gathered, commonalities and variabilities are defined and individual family members are identified. In the Domain Modeling step, requirements for each of the family members and the common requirements are analyzed using various object-oriented models. In the Domain Framework Development stage a domain framework is developed using several process models and architectural models.

Sherlock has several activities that are required for product line requirements engineering. The Domain Definition stage is equivalent to requirements capturing where requirements are gathered from domain experts, market analysis and other sources. Market analysis is a very good technique for identifying the current and future market trends because, this will have a great influence on the features and products selection in the family. It does not use any specific technique for identifying common and variable requirements. Product line requirements and potential member products are chosen according to the organizational strategies. Once the products and requirements are identified, they are analyzed using object-oriented analysis techniques. Sherlock provides tool support for managing each activity. There is no specific technique used for requirements specification, verification and traceability.

2.6. Odyssey-DE

Odyssey domain engineering is an object-oriented domain analysis and engineering technique that uses component-based software development techniques [9]. Odyssey DE has four development phases. They are Domain Viability Analysis, Domain Analysis, Domain Design and Domain Implementation. In the Domain Viability Analysis, the feasibility and the cost-benefit analysis of the current domain is performed. In the Domain Analysis phase, the product family is characterized and the scope is determined. In the Domain Design and Domain Implementation phases, the product family architecture is designed and implemented. An aspect unique to Odyssey DE is that various components in the domain interact through Common Object Request Broker (CORBA) protocol. In this way any legacy domain model or component residing in a distant environment can be easily integrated.

Odyssey-DE has several activities that are equivalent to activities in requirements engineering. The feasibility analysis, requirements capturing and requirements analysis are performed in the Domain Viability Analysis and Domain Analysis phases. However, it does not provide any technique for performing the feasibility analysis. It uses FODA's feature modeling technique for identifying the common and variable requirements. It does not use any technique for the selection of potential members and their features. Requirements are modeled using object-oriented techniques such as use case diagrams and class diagrams. Odyssey-DE uses certain design patterns (structured templates) to specify domain concepts. It also provides a traceability mechanism between requirements and other work products using tool support. However, the technique used for this is not mentioned.

3. Requirements Engineering Processes and Techniques

There are five stages in a typical Requirements Engineering process used for single product development [10]. They are requirements capturing, requirements analysis, requirements specification, requirements verification and requirements management. For product line development these activities are best being carried out in the early stages of DA&E. The major goals of DA&E are to identify potential member products and their requirements, to analyze these requirements and to design and implement a reusable domain framework. The above-mentioned five stages of the requirements engineering process can be used to identify, analyze, specify, verify and manage product line requirements and other product line related aspects. Several tools and techniques have been developed for each of the requirements engineering stages. The applicability of several such techniques to product line development is assessed in the following sections.

3.1. Requirements capturing techniques

In this stage requirements for the system are gathered from various sources [10], [11]. Interviews, Surveys, Market Analysis, Cooperative Requirements Capture (CRC), Soft System Methodology and Designer as Apprentice are some of the available requirements capturing techniques. For product line development, the requirements capturing techniques used must gather information about the target domain, related domains, potential customers and their requirements. Techniques such as context diagrams or concept mapping can be used to analyze the domain concepts. Information sources may include books, domain experts, market experts and existing products. Potential customers and stakeholders must be identified and their requirements collected. Existing product features and competing products features must be gathered. Any information related to product line practices such as the work products to be developed and specification templates to be used are defined. Depending on the situation any of the following techniques can be used.

3.1.1. Interviews

Interviewing is a technique that involves structured or unstructured discussion between requirements engineers and customers [11]. Interviews alone are not sufficient for requirements elicitation in product lines. There are several stakeholders involved and also the organization has to satisfy its business goals. However, interview techniques can support other requirements elicitation techniques. After analyzing domain concepts with domain experts and market experts, potential customers can be interviewed to capture information about a particular topic. Structured Interviews can be conducted if the requirements engineer already has a fairly good knowledge about a concept and wants to get clear answers from the customers to specific questions. Unstructured

Interviews can be used if the requirements engineer wants to have an open-ended discussion with the users. Also interviews can be used to communicate common and variable requirements that are already gathered by the users so that any negotiations can take place.

3.1.2. Surveys, Questionnaires and Data Mining

In Surveys and Questionnaires a list of questions is distributed to a sample population and their responses are gathered [11]. These techniques can assist other techniques in determining common and variable features of the family. For example, by collecting different users' responses for a question on the data input format required for a spreadsheet or similar application, it is possible to identify possible data input formats. Different stakeholders may have different views on a concept. Diversity in the users' responses is a very good way to identify the variations. Data Mining can reveal useful patterns in the data collected, which can be used to draw important conclusions about the customer requirements.

3.1.3. Soft System Methodology (SSM)

Soft System Methodology is an organizational approach for requirements engineering [11]. Instead of considering the technical problem alone, it tries to improve the organizational structure and the context in which the product is being developed. SSM is composed of seven activities: The Problem Situation Unstructured, the Problem Situation Expressed, building Root Definitions of relevant systems, conceptual modeling, comparison with the real world and implementing the feasible and desirable changes. SSM can be easily used for the capturing of product line requirements. In fact, the first few activities in SSM are more suitable for product lines than for individual product development. The activity Problem Situation Unstructured involves a textual description of the problem concept along with the context and the current organizational structure. In the next stage the problem concept is explained in more detail along with a rich picture. These two activities can be used for analyzing domain concepts and for identifying related domains and sub-domains. These activities may also identify potential customers. The next activity "Building Root Definition of Relevant Systems" involves the definition of candidate systems in the domain. This is a very good process for identifying potential family members and hence variable requirements. In general SSM can be easily adapted into a product line practice with very little modification.

3.1.4. Cooperative Requirements Capture (CRC)

Cooperative Requirements Capture [11] is a group session approach similar to Joint Application Design (JAD). CRC can become a very good product line practice since it supports the development of a generic product. A CRC session not only includes the customers and developers but also others who have a stake in the proposed system. The CRC process includes activities such as Problem Identification, Team Selection and Exploring Users and User Environments. In a product line approach, these activities could correspond to the domain concept analysis and identification of potential customers and their requirements. Activities such as Identifying Scope of the system can be extended to analyze commonalities and variabilities and to identify members of the family. CRC can become a good product line practice without much modification. However, the selection of the stakeholder representative and availability of domain knowledge is important for its success.

3.1.5. Market Analysis

The process of analyzing the product market for its current and future trends is called market analysis [11]. Market analysis is a very good starting point in both single product and product line development practices. Analysis of rival products, their features and current and future market trends give some insights about the potential products and their characteristics. Market analysis can reveal how the demand may change over the coming years. This would help the organization to make better selection of member products and their features. For example, if the market analysis reveals that the demand for a particular feature will increase in a few years, then the organization can decide to select that feature in all of its products. The PuLSE product line practice already has a technique called Product Map, which is used to determine product line members and their features. For this the results of market analysis along with other domain information are used.

3.1.6. Designer as Apprentice

The technique Designer as Apprentice can be used in combination with other techniques for requirements elicitation. When the problem concept is new or when there is a lack of domain expertise in the organization, engineers can work as apprentice at the customer's site. This would provide the designer with some practical knowledge about the user requirements. Using this knowledge as the starting point, designers then can analyze these requirements to identify the commonalities and variabilities. This would also help designers to identify other potential customers and products of the family.

3.2. Requirements Analysis

In the requirements analysis stage, the requirements gathered in the previous stage are analyzed and further refined. Structured System Analysis, Object Oriented Analysis, Joint Application Design, Quality Function Deployment and Focus Groups are some of the techniques used for requirements analysis. For product line development, the information gathered in the previous stage can be analyzed to identify the commonalities and variabilities, to determine the scope and to model the family. The commonalities and variabilities can be identified either by using product line specific techniques or other techniques such as Feature Modeling in FODA and Commonality Analysis in FAST. Once the common and variable features are determined, potential products are identified. Using these details, the feasibility of developing those products as a family is analyzed. Techniques similar to Product Map or Quality Function Deployment can be used for this analysis. Factors such as market advantage, customer's priority, competitive advantage and contribution to the domain can be considered for the analysis. Based on the results of this analysis, products and their features can be selected. Once the products and features are selected, they must be modeled. Modeling enhances the understandability and simplifies the design. Depending on the product line approach used suitable modeling techniques can be used. The applicability of several requirements analysis techniques is described below.

3.2.1. Structured System Analysis (SSA)

The Structured System Analysis approach is a function-oriented requirements analysis approach [11]. Structured System Analysis techniques can be used for product lines in a similar way as they are used in single product development. Object-oriented concepts such as generalization/specialization and aggregation/decomposition can be adopted to represent commonalities and variabilities in the domain in a similar way as they are used in FODA. Also special notations can be used to represent common and variable requirements. For example, in a data flow diagram, tasks corresponding to common requirements can be represented using solid lines whereas alternative or variable tasks can be represented using dotted lines. However, function-oriented approaches are not highly supportive of reuse. There is no object encapsulation, which supports modularized development. Although techniques like entity relationship diagrams are used to identify the domain entities, there is no precise encapsulation of concepts as objects. SSA does not support component-based design and development. Also systems developed using structured programming tend to be more difficult to manage.

3.2.2. Object-Oriented Analysis (OOA)

Object-Oriented Analysis techniques are techniques based on the Object-Oriented Programming concept [11]. OOA can also be used for product line requirements analysis. Product line practices such as Sherlock and Kobra use OOA for analysis. In both Sherlock and Kobra use case diagrams and class diagrams are used for modeling and entities corresponding to the common requirements are represented using solid lines and those corresponding to the variable requirements are represented with dotted lines. In object-oriented approaches variable requirements can be easily represented using the concept of inheritance. Variation points and their possible values can be represented using the super-class and sub-class relationship.

The major advantage of object-oriented programming is to enable reuse and maintenance. The encapsulation of attributes and behavior in objects supports modularized design and development. By decomposing the entire system into a collection of objects, many of the components can be reused without too much effort. Also smaller components are easier to maintain than a single function-oriented system that has everything in one bundle. Object-oriented approaches tend to better support reuse than various other approaches [12]. In order to reduce the cost and time to market, a product family must have a majority of the artifacts reusable across the

family. Therefore approaches used in the product line practice should support reuse. Hence object-oriented analysis is a good approach for requirements analysis in product line practice.

3.2.3. Joint Application Design (JAD)

JAD is a group session approach that involves users in the system design [11]. Since the product line practice involves several different stakeholders, a group session approach similar to JAD is a good approach to analyze product line requirements. A JAD session can be conducted in a similar way as it is conducted in single product development. The session leader can guide the session. However, it is important to identify all potential user representatives. Activities such as defining high-level requirements and bounding system scope can be extended to identify common and variable requirements and the scope of the family. All the stakeholders can participate in the decision-making process. However, the original JAD approach uses Structured System Analysis techniques for the system design [11]. Structured system analysis techniques are not very suitable for product line practices as described in section 3.2.1. Instead of SSA techniques, it should also be possible to use OOA techniques. If the facilitator and other participants have the necessary skills and if appropriate techniques are used, JAD can become a very good technique for product line requirements engineering.

3.2.4. Quality Function Deployment (QFD)

QFD is an approach for translating customer requirements into appropriate technical requirements [11]. In addition to relating the technical features with customer requirements, it considers product features of the competition, customer priorities and correlation between the features. In a product family, there is more than one product to be analyzed. In this case QFD must be modified to include all the products in the family. If needed, QFD can be applied to each member of the family separately. In such cases it will be difficult to handle the common requirements. However, by properly analyzing the results for each product, it is possible to determine the scope and features of the family. PuLSE technology has a similar technique called Product Map for product line scoping. In a Product Map, all the products and their characteristics are related using a matrix. Competing products are also considered in this matrix. Based on the subjective evaluation of the relevance and priorities, potential members and their requirements are selected.

3.2.5. Participatory Design

The Participatory Design approach allows designers and users to work together [11]. Both parties learn from each other. In a product line there are more than one product and customer. Work practices of each customer are different. Therefore it requires a lot of coordination between designers and various customers. If designers and customers can work together, it is possible to come up with a design that satisfies all the customers. In addition to this, working with different customer can lead to the resolution of requirements conflicts. Hence this is a very good approach for designing the product family where all the customers are involved.

3.3. Requirements Specification

In the Requirement Specification stage requirements are properly documented for future reference using certain specification formats or languages. Software Requirement Specification and Formal Methods are two techniques in use for requirements specification. For product line development, if the product line approach used already has a domain specification language or template, it can be used to specify product line requirements. Organizations can also use their own specification templates or some standardized Software Requirements Specification (SRS) document structures to specify product line requirements. If the organization already has a domain specification language, then it is easier to use it. If SRS is used to specify the requirements, a SRS document is developed for the family as well as for each family member. While writing the family SRS, all the common and variable requirements must be documented. All the possible values for each variation point must also be documented. For an individual product, the SRS should contain common and specific requirements that belong to it. Structured document templates such as extensible mark up language (XML) can be used to specify product line requirements [13]. Since XML allows hierarchical representation of data, common and variable requirements can easily be represented.

3.4. Requirements Verification

Requirements are verified to check for their completeness, precision and suitability in the requirements verification stage. Formal Reviews, Prototyping and Requirements Testing are some of the techniques used for requirements verification. A product family has more than one product and most of the requirements are common across the family. Any defect or misunderstanding in such requirements would affect the entire family. There are requirements that are specific to each product. Hence requirements verification is very important for product lines. Requirements verification can be conducted in a similar way as they are conducted in single product development. All the verification techniques used for a single product can also be used for product lines.

3.4.1. Requirements Reviews

Requirements Reviews are techniques that check system requirements for completeness, relevance and precision [10]. They can be either formal or informal. Formal reviews include a group session to verify the requirements. Informal reviews involve a discussion between the requirements engineer and the customer. Since there are several products involved Formal Reviews have to be conducted to verify product line requirements. The review team should include domain experts, requirements engineers, customers and stakeholders. Review can be conducted as a group meeting facilitated by the requirements engineer. At first the product family in general including all the common and variable requirements are considered for review. All the comments must be documented. Any changes required must be recorded. Next, requirements for each product in the family are reviewed. The reviewing process should also ensure that the family and its member requirements are properly mapped. Informal reviews can be conducted between the requirements engineer and customers to validate product specific requirements.

3.4.2. Prototyping

Prototyping is a technique during which products are partially implemented in order to learn more about certain problems or to demonstrate that certain features are working as intended [10]. Requirements for a product family can be verified by developing prototypes. In this case the prototype representing the common features can be reused for all the members across the family. For example, a prototype developed to illustrate the user interface facility of a system can be reused for all products.

3.4.3. Requirements Testing

The process of testing the product against each requirement is called Requirements Testing [11]. Product family requirements can also be verified by defining test cases for each requirement. While defining the test cases it is possible to unearth some of the defects in the early stage of the development. In the case of product line development test cases must be defined for both common and variable requirements.

3.5. Requirements Management

Requirements management ensures that all the requirements engineering activities are well organized and all the end products are properly documented and traceable. In product lines, requirements management is much more complicated since there are several related products and activities that are used to identify and characterize them. In the requirements capturing stage, requirements management has to document all the information collected and make sure that they are readily available. It also should ensure that all the processes and products to be used in other stages are defined. It should ensure that all the change requests are properly handled. In addition to this, requirements management has to make sure that there is a proper traceability mechanism between requirements and other domain artifacts and between the family and its member products. All the above activities can be performed with proper tool support. If document templates such as eXtensible Markup Language (XML) are used for specifying the requirements and other domain artifacts, dependency between them can be easily represented using XML links. There are tools and techniques that can parse data represented in XML documents. Using XML link documents traceability between the family requirements and other family artifacts or between the family and individual product requirements can be easily maintained.

4. Conclusion

This paper discussed requirements engineering for software product lines. It introduced requirements engineering and software product line practices. Various product line practices were briefly described. It also reviewed requirements engineering practices used in many product line practices and described how single product requirements engineering techniques can be applied to product line practices. Although many requirements engineering technique can be used for product lines, organizations can select techniques that best suite their context.

From the analysis, it is clear that most of the existing product line practices do not completely address the area of product line requirements engineering. Due to the complex and extensive nature of product line development, it is essential to have a systematic requirements engineering process. In our future work we will be proposing a requirements engineering methodology for products lines. This approach uses an aspect-oriented programming approach for identifying common and variable requirements and Product Maps for determining the scope and characteristics of the family. It uses XML as a specification template to specify requirements and maintain the dependency relationship between requirements and various other domain artifacts. This approach needs tool support that can read and write XML documents.

5. References

- [1]. Software Engineering Institute (SEI): **A Framework for Software Product Line Practice - Version 3.0**. Carnegie Mellon University.
- [2]. Software Productivity Consortium Inc. **Synthesis: Reuse-Driven Software Processes Guidebook**. Reuse-Driven Development Environment (RDE) Product Line Project, Software Productivity Consortium. 1996.
- [3]. Gupta, Neeraj.K, Lalit Jategaonkar Jagadeesan, eleftherios E.Koutsofios, David M. Weiss. **Auditdraw: Generating Audits the FAST Way**. Lucent Technologies/Bell Laboratories Software Production Research Dept, AT&T Research, Network Services Research Lab, New Jersey. U.S.A. 1995.
- [4]. David, M.Weiss. **Commonality Analysis: A Systematic Process For Defining Families**. Lucent Technologies Bell Laboratories. 1997.
- [5]. Kang.C.Kyo, Sholom, G.Cohen, James, A.Hess, William, E.Novak, and A.Spencer Peterson. **Feature- Oriented Domain Analysis (FODA) Feasibility Study**. Technical Report: CMU/SEI-90-TR-21. Software Engineering Institute. Carnegie Mellon University. 1990.
- [6]. Joachim Bayer, Oliver Flege, Peter Knauber, Roland Laqua, Dirk Muthig, Klaus Schmid, and Tanya Widen. **PuLSE: A Methodology to Develop Software Product Lines**. Fraunhofer Institute for Experimental Software Engineering, Germany and Lucent Technologies Software Product Line Engineering Laboratories, U.S.A. May 1999.
- [7]. Colin Atkinson, Joachim Bayer, and Dirk Muthig. **Component Based Product Line Development – The Kobra Approach**. Fraunhofer Institute for Experimental Software Engineering, Germany. 1999.
- [8]. Succi,G. SENG609.08 Class notes on **Sherlock: An Object Oriented Framework for Domain Analysis and Engineering**. The University of Calgary, Canada. 1999.
- [9]. Regina, M.M Braga., Claudia,M.L.Werner., and Marta,Mattoso. **Odyssey: A Reuse Environment based on Domain Models**. Computer Science Department, Federal University of Rio de Janeiro, Brazil. 2001.
- [10]. Sommerville, I. Kotonya, G. **Requirements Engineering: Processes and Techniques**. John Wiley & Son Ltd. 1998.
- [11]. Macaulay, L. A. **Requirements Engineering**. Springer- Verlag. 1996.
- [12]. Jacobson Ivar, Griss Martin, Jonsson Patrik. **Software Reuse Architecture, Process and Organization for Business Success**. ACM Press New York. 1997.
- [13]. World Wide Web Consortium (W3C). **Extensible Markup Language (XML) 1.0** (Second Edition) W3C Recommendation. 6 October 2000.
- [14]. Mark Ardis, David Weiss. **Commonality Analysis: Principles and Practice Introduction and Overview Notebook**. Lucent Technologies. 1997.