Session Report on Product Issues in Product Family Engineering

Peter Knauber[1], Steffen Thiel[2]

[1]Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6, D-67661 Kaiserslautern, Germany
peter.knauber@iese.fhg.de

[2]Robert Bosch Corporation
Corporate Research and Development, Software Technology
P.O. Box 94 03 50, D-60461 Frankfurt, Germany
steffen.thiel@de.bosch.com

Abstract. This report gives an overview of the session on product issues of the 4th International Workshop on Product Family Engineering. It briefly sketches the issues presented in the technical session and summarizes the results and open issues of the subsequent discussion session.

1 Introduction

This report summarizes the session on product issues of the 4th International Workshop on Product Family Engineering (PFE-4) that was held on October 3-5, 2001 at the European Software Institute (ESI) in Bilbao, Spain. The goal of the workshop was to bring together professionals from academia and industry to exchange ideas and experiences, to identify obstacles and propose solutions in the field of product family engineering.

The remainder of this paper is organized as follows: Section 2 sketches the topics that were addressed in the technical program of the session on product family issues. In Section 3, the topics and results of the workshop discussion that took place after the technical program are summarized. Open issues identified during the presentations as well as during the discussion are given in Section 4. Finally, Section 5 concludes with a short summary of this paper.

2 Session Topics

The presentations of the product session can roughly be organized in three practice areas: scoping, variability management, and derivation and evolution. A short summary of the presented topics is given next.

2.1 Scoping

In the context of product family engineering, scoping is the activity that determines the boundaries of the product families. The result of this activity is a documentation of the scope. The scope is a description of the products that will constitute the product family or that the product family is capable of including. At its simplest, the scope may consist of an enumerated list of product names. More typically, this description is cast in terms of the items that the products all have in common, and the ways in which they vary from one another. The description might include features or services they provide, performance or other quality attributes they exhibit, and platforms on which they run.

For a product family to be successful, its scope must be defined carefully. If product members vary too widely, the core assets will be strained beyond their ability to accommodate the variability, economies of production will be lost, and the product family will degenerate into the old-style one-at-a-time product development effort. If the scope is too narrow, the core assets will not be built in a generic enough fashion to accommodate future growth, and the product family will stagnate. This means that economies of scale will not be realized; the return on investment will not materialize.

In his presentation, Paul Clements from the Software Engineering Institute (SEI) highlighted the importance of scoping for the overall success of product family development [3]. He outlined essential issues associated with the scoping activity such as adequate level of precision for a scope definition, specific techniques to support scoping, and the relation of scoping to other product family activities. In particular, he emphasized the importance of pro-actively adjusting the scope to take advantage of new market opportunities. He mentioned CelsiusTech as one example of a company that has experience in and the skills for pro-active scoping. Clements pointed out that CelsiusTech was able to expand its business from naval applications into ground-based air defense systems. They recognized that a ground-based air defense system is very similar to a ship-based system without the ship. By reusing the abstractions inherent in the naval systems architecture, CelsiusTech was able to quickly build the new architecture, taking 40% of its components directly from the ship system asset base. Just by understanding their scope, and then shrewdly expanding it, CelsiusTech was able to enter an entirely new (but related) business area in very short time.

Klaus Schmid from Fraunhofer Institute for Experimental Software Engineering (IESE) presented an initial model of product family economics, which integrates characteristics of the software development process with market aspects [7]. The ultimate goal of constructing such an economic model is to support the determination of an “optimal” scope. His model takes factors into account that influence economical issues for product family development. In a first step, factors like project constraints, quality attributes, and market conditions must be considered. It is possible for the development organization to at least partially influence the market conditions: product quality, development schedule, and product price may have a strong influence on the market and thus on the revenues from the products. Especially the latter two of these influencing factors, the development schedule and the product price, can be optimized

by adopting a product family approach. On the other hand, these two factors and the product quality influence each other. Klaus Schmid pointed out that in order to develop a comprehensive model for product family economics, higher-level models are needed. These models then would take into account financial theory, valuation of uncertainty (from option models), and feedback cycles (the products offered and sold change their respective market). The model to be chosen in a specific situation depends on the concrete goals that shall be addressed by the corresponding product family.

2.2 Variability Management

When developing software product families instead of single systems, variability is key along two dimensions: variability in space and variability in time. Variability in space reflects the need to handle alternative variants of products (and the assets they are composed of) in parallel. This holds for the complete development life cycle, that is, planning, development, testing, and maintenance of these assets.

Variability in time can be understood and managed in a similar way as for the development of individual systems. The only difference is that for product families it may be necessary to merge asset variants that have originally been developed specifically for different (parallel) products.

In his presentation, Klaus Pohl from the University of Essen emphasized that variability is one of the main concepts for product families [2]. It increases flexibility but also complexity and cost of the software. On one hand, having the right amount of variability provided by the software allows to tailor it to specific customer's needs easily. Variability also allows for customization of products in a way that it fits into different application contexts. If, on the other hand, the software is planned with too much variability or the variability points are bound too late during the development process, the complexity of the software may become too high, and, due to this high complexity, the development costs may increase. Thus, the benefits from being able to customize the software efficiently might stay behind the upfront investment in the development. If there is too little variability or if the variability points are bound too early, the flexibility is decreased. This may cause problems in customizing the features for the planned products. However, Klaus Pohl also mentioned the lack of appropriate tool support for the management of variability points, their dependencies, and their binding times.

Andreas Reuys from the University of Essen addressed the problem of considering variability when 3rd party components are selected during product family development [6]. The selection of components off-the-shelf can take place during domain or during application design. For an appropriate selection, three facets must be taken into account: the variability of the product family, architectural concerns, and the requirements defined for the product family. The right component must fit with respect to all three facets. When evaluating a component for its fitness, one can observe an interplay between components and facets: selecting a component can

influence the family under design (or one of the products derived from it) in various ways. In order to control this influence, Andreas Reuys distinguished high-level vs. low-level component selection. During low-level component selection, constraints from the three facets should be considered as given, whereas during high-level component selection, the facets can be influenced in several ways: with respect to requirements, the evaluation of components can help to identify new or to clarify existing requirements, to resolve ambiguities, or to define more detailed requirements. Concerning variability, component evaluation can help to identify new or to clarify existing variation points, to identify new variants, or to determine concrete binding information for variation points. With respect to architectural concerns, component evaluation can help to adapt archetypes in order to refine architectural styles, or determine good patterns and constraints. The CoVAR (Considering Variabilities, Architecture, and Requirements during Component Selection) process was designed to allow component selection in a systematic way.

2.3 Derivation and Evolution

In order to resolve the variability in space (see above) in product families, processes for derivation of products from the family are necessary. These have to go hand in hand with the processes for planning, designing, and maintaining the variability within product families. Especially evolution within the domain influences the amount of variability that should be provided by the product family infrastructure. Alessandro Maccari and Rob van Ommering proposed ways to manage these evolution aspects in very large, inhomogeneous, and sometimes fast-moving domains.

In his presentation, Alessandro Maccari from Nokia Research Center explained the process for maintaining architectural documents that is implemented at Nokia Mobile Phones [5]. Documents used in this process are the reference architecture document, the configuration architecture document, and the product architecture documents. The first two of these are maintained actively. The reference architecture document contains all information required for adding components to the system. It describes architecturally significant requirements, architectural rules (currently captured using UML), the communication infrastructure, that is, the relationships among the components, and the runtime architecture which describes the processes and their interactions with each other and the operating system. The configuration architecture categorizes features according to their respective domain, with special emphasis on commonality and variability. When configuring concrete products, these features are mapped to the new products. The maintenance process for these two documents is a combination of forward and reverse engineering. The reverse engineering activity proved to be necessary because sometimes the real system conflicts with rules of the reference architecture. Experience shows that a well-defined reference architecture is a prerequisite for doing a good reverse architecting job.

Rob van Ommering from Philips Research Laboratories presented an approach to manage architectures for so-called product populations [8]. Product families are

typically built around products with many commonalities and few differences. Contrary to this, he defines product populations as addressing many commonalities but also many differences of their respective products. His proposed way to handle this situation is to combine top-down product family planning with bottom-up component development. In this way, the reference architecture for the product population helps to ensure compatibility of the components in time and space. Subsystems are developed by smaller teams that know about the final products to be expected within the next six months. An architecture team plans their integration depending on which features can be provided by which package and are needed by which other package. They are supported by a special infrastructure: each product development team publishes their requirements in a certain format on the intranet; specific tools can then be used to get an overview and to identify commonalities among the products. In this way, it becomes possible to design and develop subsystems that can be used in more than one product.

3 Discussion Topics and Results

The presentations about product issues in product family engineering were followed by a discussion about important topics and ideas which were raised during the presentations. The following three focal themes could be extracted from this discussion: variation and cost estimation, development and organization, as well as people and social aspects. Next we provide a short summary of the results of this discussion.

3.1 Variation and Cost Estimation

Concerning the first topic, variation and cost estimation, there was a discussion about the possibilities to quantify the cost of introducing a (new) variation point into a product family architecture more precisely. The workshop participants agreed on the fact that a precise estimation (e.g., in terms of money or effort) is very difficult since the introduction of variability does not only depend on technical product factors but also on the company's individual business goals, its position, culture, and success in the market that is addressed by the product family, the evolution of the market and related technologies, the legal and political situation for this market, as well as the dominance and aggressiveness of competitors. However, during the discussion some ideas arose that could help in deriving such estimations. These ideas touched technical as well as non-technical areas.

One argument, for example, was that in order to cope with non-technical aspects of the products, it is strongly recommended to include stakeholders from marketing and sales department in the estimation process. It was seen as valuable to collect opinions from as many experienced people as possible in order to lower the amount of distracting quantities. Economic models as introduced in the presentation of Klaus Schmid from Fraunhofer IESE [7], for example, were recognized to be beneficial and

may be used to support this process. This would allow for a more focussed collection of criteria and factors that influence the estimation. It was further argued that, although models, could not replace experienced people, they provide a vehicle for communication. Good models were seen as a means to guide people in keeping an eye on the important issues.

Furthermore, to address the estimation problem within product families at a more general level, the participants proposed to adopt and refine well-known (single-system) estimation models, such as the COCOMO (Constructive Cost Model). COCOMO, first published by Barry Boehm in 1981 [1], is a model that allows one to estimate the cost, effort, and schedule when planning a new software development activity. However, the discussion about cost models uncovered a lot of open issues with respect to measuring the benefits of variability (e.g., how to gather the input factors, how to keep cost models up to date etc.). As a result, the participants agreed on the fact that metrics for estimating a product family's benefits remain a challenging topic that must be addressed in more detail in future research.

3.2 Development and Organization

The discussions about development and organizational issues has concentrated on two main questions:

- Can it be justified to develop a product family from scratch?
- Does organizational size have an impact on the development?

Concerning the first question, the participants agreed on the fact that building a product family in a "green field" approach is a critical task that needs careful planning and a strong management. Dealing with the "right" scope of the product family, as motivated by the presentations of Paul Clements [3] and Klaus Schmid [7], becomes essential. The product family members as well as the business goals (e.g., how, where, and when to make money with those products) and risks associated with the effort need to be described as detailed as possible. Although there might be the decision to start the effort from scratch, an activity that is worthwhile being performed is to seek for legacy systems that "fit" to the scope under definition. Usually, a company does have legacy products, demonstrators, or prototypes that share a lot of characteristics (e.g., features) of the products to be developed. This knowledge (including that of the experts that were involved in the development of those products) should be considered during the early phases of the product family development effort, although the planned product members might address different or additional domains or business goals. The analysis of legacy information may also include seeking through the architectural documentation or, if such documentation is not available, architectural recovery. The goal is to understand the fundamental design decisions made for these products since they may also be important for the product family design. However, one has to be careful that the implementation of the legacy products does not drive the design activities too much since they are normally based on different (i.e., past) business

goals. On the other hand, some of the fundamental design decisions may provide valuable insights into solutions that can be applicable in a refined form for the new products as well.

The second question addresses the suitability of organizational structures for developing a product family. Organizational structure refers to how an organization forms groups for the various responsibilities inherent in a product family effort. All organizations have a structure, even if it is only an implicit one, that defines roles and responsibilities for to the organization's primary activities. Particular organizational structures are chosen to accommodate business goals and directives, culture, nature of business, and available talent. The organizational structure reflects the division of roles, responsibility, and authority.

In the discussion the participants especially focussed on the problem of organizational size. Some people shared their experience about this topic. The results can be briefly summarized as follows: The larger the organization, the more discipline is necessary to manage it and keep things going, the more complicated it is to propagate decisions and change requests, and the higher are the investments.

3.3 People and Social Aspects

The last major discussion topic of this workshop session was concerned with people and social aspects. The participants talked about the different views, cultures, and background of typical stakeholders involved in a product family project. For example, marketing people have fundamentally different responsibilities than software developers. Whereas the people responsible for the development of a product are concerned with technical issues like how to realize the safety and performance mechanisms documented in the architecture, marketing is more concerned with issues like delivery dates, cost, essential features for the customers, and prototype versions that allow to demonstrate features before delivery. In summary, the different stakeholders of a product family effort have different goals or “scenarios” in mind which, for their opinion, are the most important ones for the success of the project. The participants agreed on the fact that each of these goals should be carefully considered and evaluated if they are reasonable. Generally, the goals of different stakeholder groups are conflicting, so trade-offs are required in order to obtain a feasible solution for the effort. However, the decisions where trade-offs have to be made should be discussed in a collaborative workshop with all stakeholder representatives. Such a prioritization of requirements and project constraints would help to significantly reduce (or, ideally, avoid) conflicts during later development phases. The design decisions that usually have to be made frequently in later phases can then be aligned and harmonized with the project goals discussed and decided on earlier. This would contribute to a better understanding (rationale) and traceability of such decisions. It would also help to avoid inappropriate decisions (e.g., concerning the architecture) that may conflict with business goals, for example.

Another interesting issue that was discussed by the participants was the fact that project goals and, together with these, requirements and development constraints change during a product family effort. This is a normal situation in real-world projects. The question was, how to cope with such changes. After discussing several approaches, the participants worked out five basic steps that should be considered in this context: (1) expect that changes may happen for your effort and try to identify them as early as possible; (2) manage change requests systematically (e.g., name a stakeholder responsible to gather change requests); (3) organize a workshop with the stakeholders whose responsibilities might be affected by a specific change; (4) within the workshop, analyze and prioritize the change request with respect to importance and its feasibility within the current development status; and (5) make a decision for the change requests (e.g., satisfy change and make tradeoff decisions, postpone change, or reject change as irrelevant). Of course, these decisions should then be fed back to the product family development project.

4 Open Issues

The presentations during the sessions and the discussion afterwards identified a couple of open issues in product family engineering. Some of these have already been described in the sections above; the remaining ones are summarized here.

During the discussion, the issue of bringing together stakeholders with different background and different viewpoints and making them work together arose. On one hand, everybody seems to agree that the combination of people from different disciplines like software engineering, marketing, and economics can produce better and more comprehensive results than having these groups work separately on their respective tasks and trying to fit the pieces together at the end. Misunderstandings and different expectations can be identified and avoided in the beginning of a project already, and working on common documents and models helps people to understand each other better. On the other hand, it remains an open issue how to achieve such an interdisciplinary cooperation and how to make it work. One was to introduce psychological aspects into the technical and business discussions.

Another open issue is the combination of mathematical models with informal estimations of people. In many environments, guesses from experienced people or rules of thumb work quite well and replace complex and sound mathematical models, for example, to estimate the effort needed to produce some software or to predict the evolution of a certain market segment. On the other hand, the respective companies depend very much on these experts. It is hard to make good predictions without them, and it is difficult to communicate their estimation results (and especially the rationales behind those estimations) to others. Whereas this is an issue for software development in general, appropriate models are even more essential for product families where a whole series of products is planned and designed at once: wrong estimates may raise in the future. The participants agreed that it would never be possible to completely replace all human experts with models and/or simulations. Especially, in difficult situations, for example, when dealing with very large or complex software

components, where models are essential to manage size and complexity, experts are needed to manage, feed, and adapt these models. It remains an open problem to find the right balance between rational models and emotional decision-making, that is, to locate the borderline between human expertise and supporting formal models.

Related to this modeling issue, future research needs in economic modeling, especially cost modeling, were discussed. First, there is the underlying problem of determining the right metrics for measuring the benefits of reuse, providing a certain range of variability etc. Based on the solution of these problems, approaches known from financial theory, for example, approaches to deal with uncertainties, can be transferred and used to define optimal product portfolios and thus the requirements for product families. Another problematic topic is the influence of products on the market: the development of a product family is based on a certain expectations about how the market addressed by the family will evolve in the future. However, not only product releases from competitors but also releases of own family members may heavily influence that market, with the consequence that the basic assumptions from the start of the project are no longer valid. Appropriate models for market behavior simulation are needed in order to assess the overall benefits of the family approach. Related to these model-building problems is the lack of validation for existing models and approaches: hardly any of these have been documented and validated in a reproducible form.

After several years of research in product family approaches, there is still only a limited tool support available for creating and managing product family assets in an appropriate manner. Also, the representation and binding of variability is not addressed satisfactorily. Adaptations of existing tools that make them suitable for managing assets (e.g., common and variant requirements) are time-consuming and hard to maintain (e.g., in new tool releases). Most commercial and research tools do not work together properly, so time-consuming workarounds have to be made. The audience agreed that tool support is important if not essential for product family development, especially in large projects. On the other hand, having the right tools does not guarantee successful software development.

6 Summary and Conclusions

In this paper we have reported about the session on product family issues that was held within the scope of the 4th International Workshop on Product Family Engineering (PFE-4). We have given a brief overview of the topics that were presented as part of the technical session. Next, we have provided a summary of the issues and results that could be gathered during the discussion session. A discussion of open problems was given in the remainder of this paper.

As one can see from the discussions in this report, there exists are a lot of promising work that addresses various problems in different areas of product family engineering. However, there are still a lot of open issues to be investigated in future research activities in order to solve the problems that arise when adopting a product family approach in an industrial context.

References

1. B. W. Boehm: Software Engineering Economics; Prentice Hall, 1981.
2. J. Bosch, G. Florijn, D. Greefhorst, J. Kuusela, H. Obbink, K. Pohl: Variability Issues in Software Product Lines; In: Proceedings of the 4th International Workshop on Product Family Engineering (PFE-4), Bilbao, Spain, October 3-5, 2001, pp. 11-19.
3. P. C. Clements: On the Importance of Product Line Scope; In: Proceedings of the 4th International Workshop on Product Family Engineering (PFE-4), Bilbao, Spain, October 3-5, 2001, pp. 69-77.
4. P. Clements, L. Northrop: Software Product Lines: Practices and Patterns; Addison-Wesley, Reading, MA, 2001.
5. A. Maccari, C. Riva: Architectural Evolution of Legacy Product Families; In: Proceedings of the 4th International Workshop on Product Family Engineering (PFE-4), Bilbao, Spain, October 3-5, 2001, pp. 63-68.
6. K. Pohl, A., Reuys: Considering Variabilities During Component Selection in Product Family Development; In: Proceedings of the 4th International Workshop on Product Family Engineering (PFE-4), Bilbao, Spain, October 3-5, 2001, pp. 21-36.
7. K. Schmid: An Initial Model of Product Line Economics; In: Proceedings of the 4th International Workshop on Product Family Engineering (PFE-4), Bilbao, Spain, October 3-5, 2001, pp. 37-47.
8. R. van Ommering: Roadmapping a Product Population Architecture; In: Proceedings of the 4th International Workshop on Product Family Engineering (PFE-4), Bilbao, Spain, October 3-5, 2001, pp. 49-61.