PuLSE-DSSA – A Method for the Development of Software Reference Architectures

Jean-Marc DeBaud
Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6
D-67661 Kaiserslautern, Germany
debaud@iese.fhg.de

Oliver Flege
Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6
D-67661 Kaiserslautern, Germany
flege@iese.fhg.de

Peter Knauber
Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6
D-67661 Kaiserslautern, Germany
knauber@iese.fhg.de

1. ABSTRACT

Software architectures are one of the most important assets developed and used in the software development life-cycle. They are an appropriate means for specifying a system, understanding it, and communicating its high-level static and dynamic aspects to the various stakeholders. In the context of software product lines, software architectures are even more important because all members of the product line are meant to share the same reference architecture. Nevertheless, almost no approaches exist for the systematic development of reference software architectures. This position paper presents PuLSE-DSSA, a method for the systematic and iterative development of reference architectures for software product lines.

1.1 Keywords

software reference architecture, architecture development, software product lines

2. INTRODUCTION

Software architectures play a key role in the software life-cycle. During system development, they can be used for specifying the static and dynamic structure of an upcoming system and for guiding incremental development; during maintenance, they help to ensure conceptual integrity. Throughout the life-cycle, they facilitate communication among the various stakeholders. Despite the significance of software architectures, up to now only few systematic and well-defined methods exist for architecture development and analysis. Probably the best-known method for the analysis of architectures is the Software Architecture Analysis Method (SAAM) which has been developed at the Software Engineering Institute [2].

For product lines, architectures are even more important than for single systems. Contrary to a set of similar but one-at-a-time developed systems, the members of a product line share a common reference architecture which ensures their conceptual integrity. Developing systems based on instances of this common architecture implies a high potential for reuse and related benefits like increased quality, cost reduction, decreased time-to-market, etc. However, due to the required degree of flexibility, product line architectures are even more difficult to conceptualize than those for individual systems.

At the Fraunhofer IESE, we are currently developing a product line methodology for software systems called PuLSE (Product Line Software Engineering). PuLSE encompasses six technical components that cover all aspects of product line development and evolution. One component, PuLSE-DSSA (domain-specific software architecture), is a framework for developing product line reference software architectures. Although it addresses a number of issues specifically related to product lines, this framework can also be used for the development of single system architectures.

Section 3 presents a high level view of the PuLSE-DSSA process model. Its resulting work products are described and prototyping aspects of architecture development are discussed shortly. Section 5 summarizes this position paper and highlights some lessons we learned from the first two PuLSE-DSSA applications in currently running industrial transfer projects.

3. PuLSE-DSSA – THE PROCESS

3.1 Key Elements

Before we present an overview of the architecture development process in section 3.2, we will describe its key elements.

Development and Analysis Aspect. PuLSE-DSSA provides both a framework for systematic development of a reference architecture, and a way to analyze the quality of software architectures with respect to specific properties.

Scenarios. PuLSE-DSSA uses scenarios similar to the task scenarios in SAAM [1]. For the purpose of PuLSE-DSSA, scenarios are categorized as either generic or property-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ISAW3 Orlando Florida USA
Copyright ACM 1998 1-58113-081-3/98/11...$5.00

related.

Generic scenarios represent architecturally significant functional requirements of a product line. Each of them may either cluster and represent a range of variability within the domain, or it may represent a common functionality of all product line members. As in SAAM, generic scenarios are represented using textual descriptions. Although this is likely to require a transformation of input workproducts (see section 4), it is nonetheless necessary to consolidate their respective information as well as to provide it in a unified form for the architecture development process.

Property-related scenarios focus on domain-independent quality aspects such as coupling and cohesion, performance, and extensibility. Because of their domain-independence, they can be provided in a library where they are grouped in categories and stored together with models on how they should be applied.

Configuration Model. During the development of the reference architecture, certain decisions have to be made that are not driven by the domain itself (e.g., use of different database systems or of alternative implementation strategies). These decisions may introduce domain-independent variabilities and therefore have to be captured explicitly in the PuLSE-DSSA configuration model.

Traceability. During architecture development, traceability links are established from components of the product line model (see section 4) to scenarios, and from scenarios to components and connections in the reference architecture. These links are maintained during the lifetime of the product line and used to keep the product line model and the reference architecture consistent. Of course, the full potential of traceability information can only be exploited if PuLSE-DSSA is performed within a full PuLSE development life-cycle (see section 4).

3.2 Process Overview

The basic idea of PuLSE-DSSA is to incrementally develop a reference architecture guided by generic scenarios that are applied in decreasing order of architectural significance.

At first, generic scenarios are developed using information from the product line model (or whatever workproducts are available as input). These scenarios are then sorted according to their architectural significance. A basic set of them is used to build an initial architecture. After that, the remaining scenarios are applied one by one to the current architecture candidate to refine or extend it. This leads to new candidates that are analyzed and ranked based on functional coverage and coverage of property-related scenarios. The best one(s) serve(s) as input for the next iteration step. This iteration stops after all generic scenarios have been applied. Figure 1 shows a high-level version of the PuLSE-DSSA process model

3.3 Process Description

This section describes PuLSE-DSSA in detail, the paragraph numbers correspond to the step numbers in figure 1. The process starts with step 1 to 3, then an iteration over step 4 to 7 is performed.

1 Generic scenarios are derived from the (functional and non-functional) requirements that have been determined during the modeling of the product line. Input from the modeling component of PuLSE are a set of generic storyboards with a decision model capturing domain-related decisions and other domain-specific workproducts that together comprise the domain model (see section 4).
Like the task scenarios used in the SAAM method the scenarios used by PuLSE-DSSA are textual descriptions. They represent those requirements for the system to be developed which are important enough to be considered on an architecture level.

2 For each generic scenario, property-related scenarios are selected from a library provided by PuLSE-DSSA. The library contains a list of those property-related scenarios grouped in categories together with models on how they should be applied. The scenarios allow the evaluation of aspects such as coupling and cohesion, performance, and extensibility. To convey a sensible meaning in the context of a particular generic scenario, the property-related scenarios have to be instantiated and adapted (i.e., parameterized) to this very context. Of course, not every property-related scenario can and should be applied to each generic scenario, thus they have to be carefully selected and are then attached to the respective generic scenario.

3 The generic scenarios are sorted according to their architectural significance. Scenarios representing important variability aspects should be considered first, then those addressing structural issues and at last the ones dealing with less essential functional requirements for the system (or the systems in the domain).
Being a subjective task, the sorting of the scenarios must be performed carefully and should be documented thoroughly.

4 To start with architecture development, a basic set of generic scenarios is selected that is used to create the initial components and connections of the architecture. In the following iterations, the architecture is refined and extended step by step until all generic scenarios have been applied and the architecture description has evolved to its final state.
It is important to state that PuLSE-DSSA sets no restrictions on the method actually used for the design of the architecture components and connections though it can be used for that purpose. It provides a framework that guides the logical application of the method(s) already established within the development organization. The same statement holds for step 7.

5 Depending on the selected scenarios, more than one candidate architecture may result from step 4. In this case, the property-related scenarios attached to the currently used generic scenarios are used to evaluate and rank the candidates (i.e., to identify the best one(s) under struc-

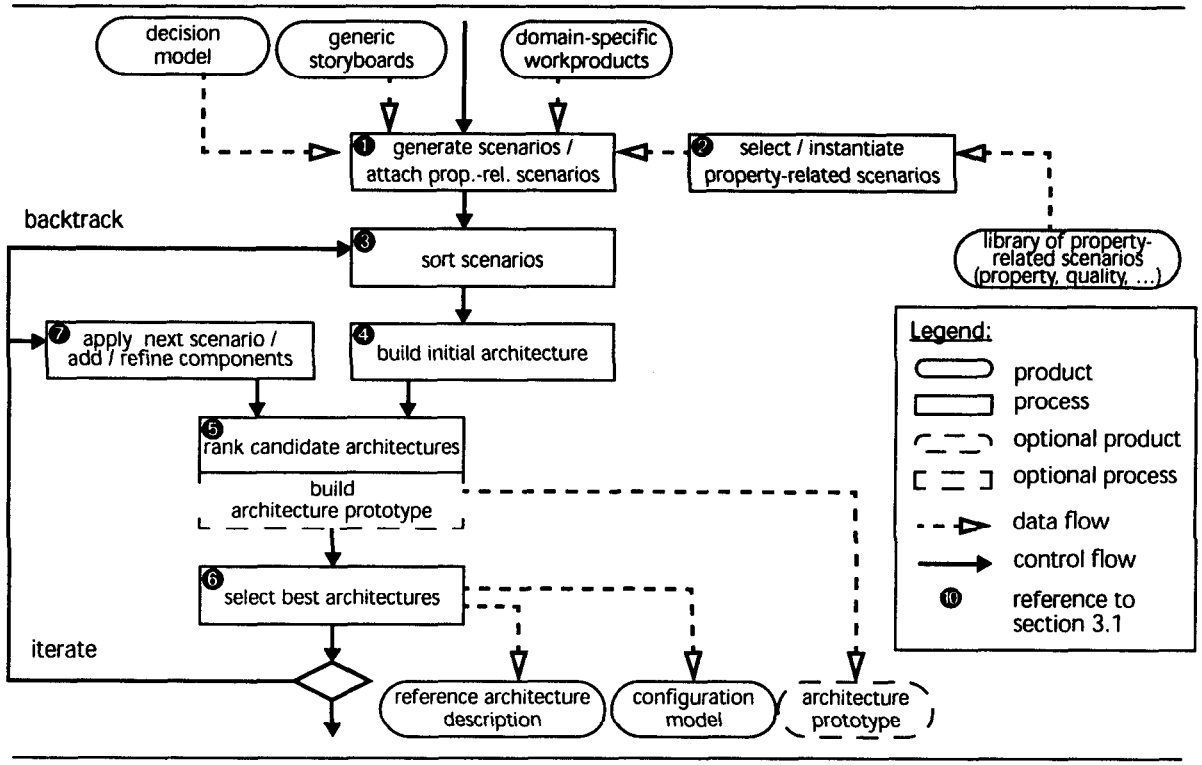

Figure 1. PuLSE-DSSA Process Model

tural aspects). To support the ranking, architecture prototypes may be developed (see section 3.4).

6 According to the ranking in step 5, the best candidate (or a set of them if no single best candidate can be determined) is selected for further evolution.
If no more generic scenarios are left, the development process is finished.

7 The next scenario from the list is applied to the selected candidate architecture(s) and the corresponding refinements or extensions necessary are determined. As a result of these steps, again more than one candidate may have been produced and the development process is continued with step 5.

We do not believe that it is possible to operationalize step 4 and 7 in a way that a "method" could replace the creativity and experience of a system architect. However, the iterative application of scenarios in decreasing order of architectural significance can help to guide and streamline the design process. Consequently, the order of scenario application has a high impact on architecture development. The wrong prioritization of the generic scenarios in step 3 may lead to inadequate architecture candidates and thus cause backtracking to the sorting step 3, where the ranking of some of the scenarios is changed. Then the architecture development can be continued starting with the architecture state that was developed by applying the unchanged part of the scenario list.

The major assets produced by PuLSE-DSSA are the description of the reference architecture and the configuration model (and, if prototyping is applied throughout the process, an architecture prototype, see section 3.4). In another workproduct, the ranking of the generic scenarios and especially the reasons for their ranking are captured. This information is used, if the scenario order has to be changed in a possible backtracking step (see figure 1). Also, the history of scenario applications is recorded to establish traceability from specific scenarios to design decisions. Whenever some of the scenarios change during the lifetime of the software system, the ramifications of these changes for the architecture can be identified using that history information.

3.4 Prototyping Aspects of PuLSE-DSSA

An important customization aspect of PuLSE-DSSA is to decide whether or not to adopt a prototyping approach and if so, to which extent.

From our experience we learned that it is very hard to develop a (product line) architecture from scratch on a purely conceptual basis (i.e., by using diagrams, informal language, and perhaps ADLs). The first, and most important, shortcoming of such an approach is that fundamental risks cannot be eliminated by just reasoning about them. Such risks are, for example, the integrability of COTS components, the compatibility of different development tools, the achievement of the required performance, and, especially for product lines, the suitability of those mechanism that allow for variability. A second shortcoming is that the comparison of architecture candidates and their subsequent ranking can not be based on

objective criteria. There are very few measures that can be applied to architecture descriptions, and for those that do exist, the significance and interpretation of the measurement results is unclear (especially if a reference architecture is considered).

If it has been decided to use prototyping, this can be done in two ways. The first approach is to develop several throwaway prototypes, where each prototype focuses on the validation of a single aspect. This allows for fast prototyping and does not impede backtracking during architectural development too much, but it is likely that certain risks are not evaluated at all and that contradicting aspects of multiple architectural decisions will not be uncovered. The second approach is to build an evolutionary prototype during PuLSE-DSSA that is then used and extended in subsequent components. It might seem as a drawback that it takes more time to build such a prototype. But the availability of a solid, stable, and validated architectural baseline at the end of PuLSE-DSSA is the best indication for the success of a project.

4. THE CONTEXT: PuLSE

The PuLSE methodology (Product Line Software Engineering) enables the conception and deployment of software product lines within a large variety of enterprise contexts. PuLSE consists of six technical components that deal with:

- Customizing: how to perform an enterprise baselining and customize PuLSE to the environment
- Scoping: how to effectively scope the product line focussing on products (i.e., not on epistemic application domains)
- Modeling: how to model the product characteristics found within the scope of the product line and explicitly denote the product family members
- Architecting (PuLSE-DSSA): how to develop the reference architecture while maintaining the traceability to the model
- Instantiating: how to specify and instantiate product line members
- Evolving and managing: how to evolve product line assets, and deal with configuration management issues as products accrue over time

PuLSE-DSSA is preceded by the modeling component and followed by the instantiating component. In the modeling component the product line concepts and their interrelationships are elicited, structured, and documented in the product line model. The workproducts used are defined during the customizing step. We distinguish between generic storyboards and other domain specific workproducts. The storyboards are used to capture relevant types of action sequences in the domain. Examples are workflow diagrams and message sequence charts. Other workproducts capture additional views on the product line. To derive product line member specifications from a product line model, a decision model is created. The decision model, the generic storyboards, and the other workproducts are passed to PuLSE-DSSA.

The instantiation component of PuLSE aims at specifying, instantiating, and validating one member of the product line. This tasks includes the instantiation of the reference architecture. Driven by the product specification and the configuration model, the architecture for the product is defined.

5. SUMMARY

This position paper gives an overview on PuLSE-DSSA, a systematic method for the development of software architectures. PuLSE-DSSA was originally designed for product line purposes but works as well for architecture development of single systems.

We started to apply PuLSE-DSSA in two currently running projects. Our first experiences indicate that task scenarios are a good means not only for the analysis (that was reported by the authors of SAAM) but also for the development of software architectures. A key step within PuLSE-DSSA is the prioritizing of the scenarios because the order in which they are ranked and then applied directs major design decisions. In the context of product line development, scenarios representing the types of variability planned for should be ranked most significant but also the early consideration of quality aspects proved to be important.

6. REFERENCES

[1] R. Kazman, G. Abowd, L. Bass, P. Clements: *Scenario-Based Analysis of Software Architecture*, IEEE Software, 11/1996

[2] L. Bass, P. Clements, R. Kazman: *Software Architectures in Practice*, Addison-Wesley, 1998.