5-2012

Feature modeling: An extended perspective to design IT infrastructure

Peter Krüger
Volkswagen AG, peter.krueger6@volkswagen.de

Feature modeling: An extended perspective to design IT infrastructure

Peter Krüger
Volkswagen AG (Germany)
peter.krueger6@volkswagen.de

Abstract

Feature modeling is a widely used method in systems engineering. Feature models help to communicate the requirements and architecture of IT systems. Applying feature models to IT infrastructure is a rare explored field in system development. We advocate using feature modeling for designing IT infrastructure to give an additional perspective on systems architecture and implement functions and properties of IT infrastructure. In this research-in-progress paper we present the idea of combining feature modeling and IT infrastructures and propose an exemplified feature model in context of IT infrastructure and its corresponding research questions. The discussion leads to functional and non-functional-requirements, feature-oriented description of architecture, and scope of IT infrastructure.

Keywords

IT infrastructure, systems architecture, feature modeling

1. Introduction

Information technology (IT) infrastructure is the data processing backbone of each organization. IT infrastructure is a set of shared, tangible IT resources forming a foundation for business applications. These, composing an IT infrastructure, are platform technology (hardware and operating systems), network and telecommunication technologies, data, and core software applications (Duncan, 1995). Figure 1 illustrates the scope for IT infrastructure mentioned in this paper.

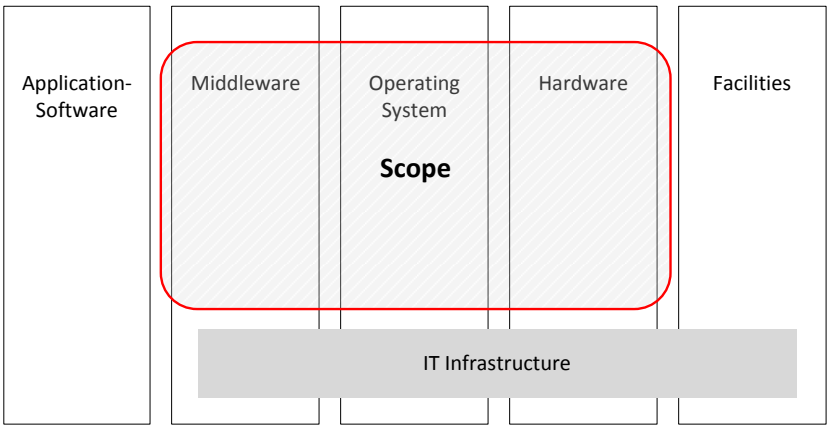

Figure 1: Scope of regarded IT infrastructure

Developing and extending IT infrastructures is a difficult issue due to heterogeneous technologies available and complex system landscapes. The realization of IT infrastructures has to be efficient with regard to costs well as time. Both criteria are essential for organizations to be competitive.
IT components are the basis of IT infrastructures and are commodities available from marketplace (Broadbent et al., 1999). Selecting and assembling these components into architectures is attended with several challenges. An appropriate documentation of the architecture is needed for stakeholder to communicate and prevent misunderstandings. The requirements, propagated by the customer, have to be interpreted by the developer and architects. After this they design the IT system by using modeling techniques to represent the design and structure of an IT infrastructure. In other engineering disciplines, like electrical engineering or mechanical engineering, a distinctive, feature-oriented view on products is established (Kossiakoff and Sweet, 2005). In the field of software engineering, feature models are an applied method to abstract the functional structure from (software) product (Kang et al., 1998). Requirements of IT infrastructure encompass different technology layers and affect software and hardware artifacts. Therefore, IT infrastructure, with elements from hard- and software, has special needs towards exposition of architecture.
The feature model is part of communication documents during the development process. In this paper we suggest to expand feature modeling to the needs of designing IT infrastructures, especially as an additional view towards systems architecture. We examine the solution idea and its research design, and elaborate an example. Intentionally, the proposal presents the ability to describe the architecture of IT infrastructures from the functional view with the assistance of feature models.

2. Background

A definition for feature from software engineering field is given by Apel and Kästner. They define a feature as a unit of functionality of a software system that satisfies a requirement, represents a design decision, and provides a potential configuration option. (Apel and Kästner, 2009) Lee argues that features are any prominent and distinctive concepts or characteristics which are visible to various stakeholders. Identifying them from externally visible characteristics of products in domain, and organizing them into a model is the activity of feature modeling. (Lee et al., 2002) Kang introduced the feature-oriented reuse method to support the software development by using reusable architectures and components (Kang et al., 1998).
Elements and their relationship to each other form a system. The systems architecture embodies information about how the elements relate to each other and suppresses details of elements to represent the system on a higher abstraction level. (Clements et al., 2011) Architectural pattern is a description of element and relation types together with a set of constraints on how they may be used. (Bass et al., 2003) Systems can have more than one structure; in fact, that is the architectural representation. For describing a system's architecture different perspectives are needed and the actual viewpoint has to be clear. Clements et al. offer a classification for common software architecture structures. They suggest three architectural structures: Module, Component-and-Connector, and Allocation. Each structure can be considered by a view, e.g. decomposition, uses, layered, or deployment. (Bass et al., 2003) The feature view of any architecture is not mentioned.
To meet the customers' wishes, requirements must be specified before systems construction begins. For that reason, a requirement is defined as something that the system or product must do or a quality that it must have. (Robertson, 1999) The discipline which deals with discovering, developing, tracing, analyzing, qualifying, communicating and managing

requirements is called requirements engineering. (Hull et al., 2011) Requirements are classified as functional requirements and non-functional requirements (NFR). The first-mentioned type determines objects what a system must do. The NFR define all the remaining claims. Robertson subsumes all properties as NFR, characteristics as well as quality attributes, which a system must have. Often mentioned non-functional requirements are performance, usability, security, cultural and political requirements, maintainability, as well as look and feel. (Robertson, 1999)

3. Idea and research methodology

As shown, the functional and non-functional requirements are used to determine the systems desired results. Steps in system development are analyzing the requirements and setting up the architecture. Finally, the architecture is designed from different views. We suggest concatenating the functional and non-functional perspective of requirements eliciting with the functional view on IT systems. This is the feature-oriented view. Hence, the feature-oriented perspective has to be extended to the field of IT infrastructure.

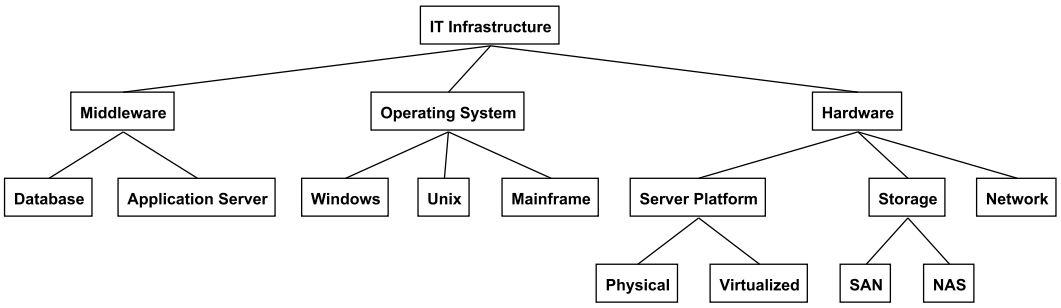

Figure 2: An exemplary feature tree without relationships representing the concept of IT infrastructure

A feature view on IT infrastructure is given in Figure 2. This exemplary illustration is simplified and only represents the tree structure without constraints or relationships. The nodes are potential features of an IT system and represent an object on a specific abstraction level. The nodes are connected with edges, in this case without explaining how each box links each other. Relationship expresses in feature models, if a feature is mandatory, optional, or alternative. A constraint is a kind of static dependency among features (Zhang et al., 2006), and changing constraint attribute of a feature can have effects on other features (Hammer, 2002) or even the whole IT systems architecture. For example, a change of CPU design of a server platform to 64-bit will directly affect the range of possible applicable operating systems.

This research applies to the approach “design science” for creating and evaluating IT solutions and its models, methods and systems (Wilde and Hess, 2007). It is a conceptual deductive analysis of finding current IT infrastructure features, transporting them into feature models, and evaluating the models. This research methodology generates suggestions for other scientists to achieve the objective. One objective is to proof the descriptiveness of IT infrastructure architecture from the feature-oriented perspective in addition to existing design descriptions. With an example, including features and constraints, we will demonstrate the use of feature models for IT infrastructures. This example is modeled as feature diagram.

4. Demonstration and discussion

Figure 2 shows a generic feature tree for IT infrastructure. In this section we will show our approach by using the example of authentication (see Figure 3). The procedure authentication confirms the identity of a person or machine for accessing data or executing actions. This procedure demands the support of several IT components. Therefore, a conceived *Authentication* feature will be refined and divided into three sub-features, *Web_Access*, *Directory_Service*, and *Policy_Service*. They have an or-relation with their parent. That means a distinct architecture contains at least one of these sub-features. The children of the sub-feature *Authentication_strong* have an alternative relationship with their parent. In this model we presume that for realization of authentication with the attribute "strong" exactly one technology must be implemented. Mandatory features must appear in all instances, in which its parents appear. In contrast optional features can appear in instances, in which its parents appear.

Features can have constraints to other features. For example, the reverse proxy "IBM Websphere" runs on operating system UNIX. Hence, there must a require-constraint which expresses the implication of the feature *UNIX* when the feature *IBM_Websphere* is selected (*IBM_Websphere* $\Rightarrow$ *UNIX*). With require- and exclude-constraints it is possible to constitute relations of features.

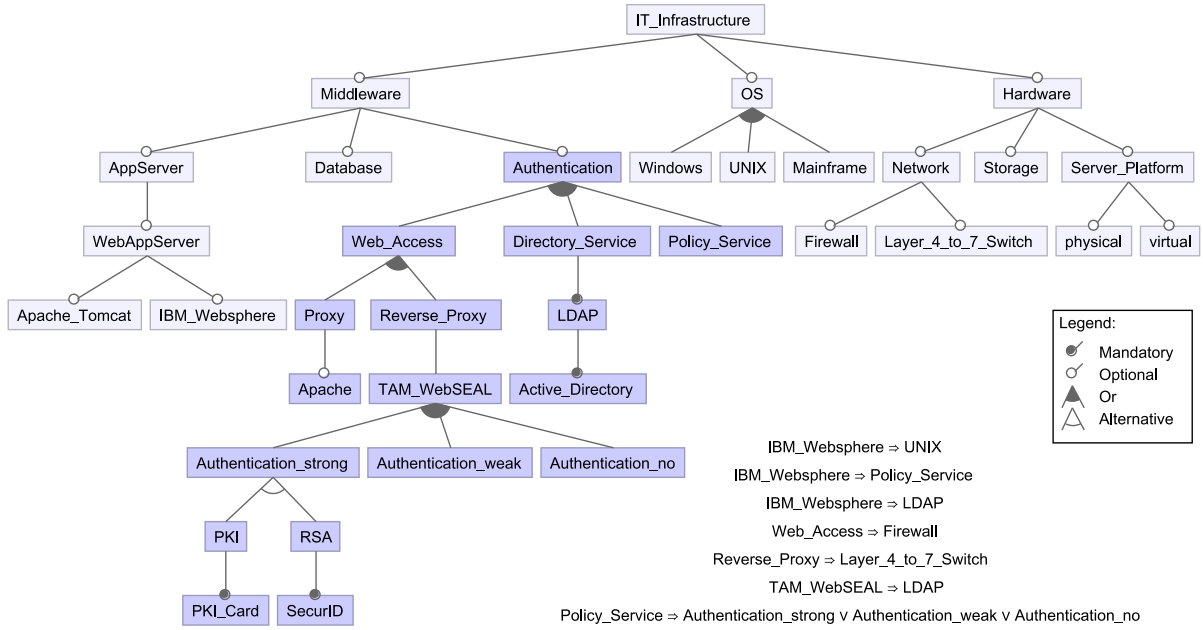

Figure 3: Feature-tree focused on the feature "Authentication"

The shown example is a basic feature model. Representing attributes in a more detailed way can be expanded by using extended feature models. Cardinalities can be represented with the model-type cardinality-based feature model, and put in the aspect of quantity. (Benavides et al., 2010) Nonetheless, the model, its features, relations, and constraints achieve the goal of an additional architectural view on IT infrastructure by using the concept of features. Predefined constraints reduce the number of possible combinations in a structured way. That means, improper combinations are prevented early and help the developer by designing the architecture. The nature of IT infrastructure with components of hard- and software is coped with feature models. Even explicit product labels don't cut the information of the diagram.

5. Conclusion and next steps

In this paper we suggested to extend the architecture view on a system by the feature-oriented perspective. We mentioned to adopt the existing feature modeling approach towards the field of IT infrastructures for realizing requirements in a more linear way. In further research, the following questions have to be answered:

- How can functional and non-functional requirements be represented in feature models? (see also (Siegmund et al., 2011))
- Are there relationships between functional and non-functional requirements of IT infrastructures?
- How can quality attributes of IT systems be documented with feature models? (The International Organization for Standardization gives advice about quality attributes; see standard ISO/IEC 25010 (International Organization for Standardization, 2011))
- What is the benefit of feature modeled IT infrastructure in context with other approaches, e.g. Attribute-Driven Design, or product line engineering (PLE)?

Especially for the product line engineering the results of applying feature models to IT infrastructures are valuable. In the context of IT infrastructure another research topic is affected. This topic is modularizing IT infrastructure with product lines. Elements of PLE can be represented with feature models like variability points, cardinalities, and constraints etc. So, this work has relevance for designing IT infrastructures architecture and also modularizing IT infrastructure.

References

Apel, S. and C. Kästner (2009) "An Overview of Feature-Oriented Software Development An Overview Development", *Journal of Object Technology*, (8)5, pp. 49–84.

Bass, L., P. Clements, and R. Kazman (2003) *Software architecture in Practice, 2nd ed.*, Boston: Addison-Wesley.

Benavides, D., S. Segura, A. Ruiz-Cortés (2010) "Automated Analysis of Feature Models 20 Years Later: A Literature Review", *Information Systems*, (35)6, pp. 615–636.

Broadbent, M., P. Weill, B. S. Neo (1999) "Strategic context and patterns of IT infrastructure capability", *Journal of Strategic Information Systems*, (8)2, pp. 157–187.

Clements, P., F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, P. Merson, R. Nord, and J. Stafford (2011) *Documenting software architectures: Views and beyond, 2nd ed.*, Upper Saddle River: Addison-Wesley.

Duncan, N. B. (1995) "No Access Capturing Flexibility of Information Technology Infrastructure: A Study of Resource Characteristics and Their Measure", *Journal of Management Information Systems*, (12)2, pp. 37–57.

Hammer, D. K. (2002) "Component-based Architecting for Distributed Real-time Systems: How to achieve composability?" in Akşit, M. (ed.) *Software architectures and component technology*, Boston: Kluwer Academic Publishers, pp. 59–98.

Hull, E., K. Jackson, and J. Dick (2011) *Requirements Engineering*, London: Springer.

International Organization for Standardization (2011) *Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models*ISO/IEC 25010.

Kang, K. C., S. Kim, J. Lee, K. Kim, G. J. Kim, E. Shin (1998) "FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures", *Annals of Software Engineering*, (5), pp. 143–168.

Kossiakoff, A. and W. N. Sweet (2005) *Systems Engineering Principles and Practice*, Hoboken: Wiley.

Lee, K., K. C. Kang, J. Lee (2002) “Concepts and Guidelines of Feature Modeling for Product Line Software Engineering”, *Software Reuse: Methods, Techniques, and Tools: Proceedings of the Seventh Reuse Conference (ICSR7)*, pp. 62–77.

Robertson, S. (1999) *Mastering the requirements process*, Harlow: Addison-Wesley.

Siegmund, N., M. Rosenmüller, M. Kuhlemann, C. Kästner, S. Apel, G. Saake (2011) “SPL Conqueror: Toward optimization of non-functional properties in software product lines”, *Software Quality Journal*.

Wilde, T. and T. Hess (2007) “Forschungsmethoden der Wirtschaftsinformatik: Eine empirische Untersuchung”, *Wirtschaftsinformatik*, (49)4, pp. 280–287.

Zhang, W., H. Mei, H. Zhao (2006) “Feature-driven requirement dependency analysis and high-level software design”, *Requirements Engineering*, (11)3, pp. 205–220.