An Integral Hierarchy and Diversity Model for Describing Product Family Architectures

Peter van den Hamer¹, Frank van der Linden¹, Alison Saunders² and Henk te Sligte¹

¹Philips Research Laboratories, Prof Holstlaan 4
5656 AA Eindhoven, Netherlands
{vdhamer,flinden,tesligte}@natlab.research.philips.com
²Philips Medical Systems, Veenpluis 6
5684 PC Best, The Netherlands
alison.saunders@best.ms.philips.com

Abstract. Formal and informal methods for describing software architectures traditionally focus on a system’s components and the interfaces between these components. They assume a design process in which the architect basically defines the architecture of a single product. If variants of this product are required, they are either handled implicitly or are defined at a later stage. Since industrial development processes which result in families of products are becoming increasingly common, there is a need for architectural models and notations in which diversity is modelled as an explicit and integral part of the architecture definition process. We believe that the use of such models can promote the overall optimisation of product families and can facilitate validation of architectural decisions. This paper presents a model which can be used to describe and manage architectures in a number of product family-oriented design processes.

1. Introduction

1.1 Product families

Industrial organisations increasingly need to develop software or software embedded in hardware systems in multiple variants. This product diversity can be necessary to accommodate different levels of functionality, regional requirements or differences in preference between market segments. A major challenge for the

† Parts of this work have been performed within the ESPRIT project 20477, ARES: Architectural Reasoning for Embedded Systems
The architect of such a product family is thus to satisfy the required commercial diversity while limiting the number of software components which need to be specified, implemented and tested.

The potential business benefits of family-oriented product development are significant: decreased development effort due to component reuse and the opportunity to optimise the overall family (global optimisation). In addition, product family development tends to go hand in hand with the validation of whether an architecture is future-proof: if one has explicitly considered the impact of various optional features, this tends to increase the likelihood that the architecture can accommodate unanticipated requirements.

1.2 Diversity-aware architecture models

Unfortunately, although many organisations employ (often implicit) forms of family-oriented product development, this does have its price. It requires the software architect to take yet another factor into account during the design of the architecture. In addition it requires the organisation to make its planned commercial diversity explicit in an early stage of the product development process.

Family-oriented product development poses thus considerable demands on the ability of an organisation to internally communicate. We therefore believe that the methods and tools used by architects to describe software architectures should explicitly accommodate the diversity or variant dimension [7] of architecture design. This supports the architect when dealing with this complex web of information. It also gives the other stakeholders the opportunity to validate the architect’s plans from the perspective of their field of expertise.

1.3 Goals of model

In this paper we describe a model for representing the architecture of a family of technically related products. To visualise examples of such architectures, we use a simple graphical notation. We want to focus on the underlying model, rather than on the various somewhat arbitrary graphical or syntactical elements of a particular notation.

With this model we attempt to satisfy the following main goals:

1. The model should cover system hierarchy or system composition as well as component diversity.
2. The model should be unambiguous because it serves as a means of communication about strategic information between persons with a wide variety of backgrounds.
3. The model should cover a broad range of family-oriented software development styles and processes.
4. Minimal training should be required to learn how to interpret the model’s concepts and notation.
5. The model should be compatible with mainstream methods for designing hardware and software systems.