

DiVA: Dynamic Variability in complex Adaptive systems

Dhouha Ayed

Dhouha.ayed@thalesgroup.com

Route départementale 128

Palaiseau, France.

Abstract: The goal of DiVA is to provide a new tool-supported methodology with an integrated framework for managing dynamic variability in adaptive systems. This goal will be addressed by combining aspect-oriented and model-driven techniques in an innovative way.

1. Introduction

Adaptive systems are able to dynamically adapt to their context. Managing adaptation of complex systems that include multiple heterogeneous interacting services is difficult since these services tend to be distributed, interdependent and sometimes tangled with other services. Furthermore, the exponential growth of the number of potential system configurations derived from the variability of each service need to be handled. The objective of DiVA is:

- 1) To provide novel build time and runtime management of adaptive systems having co-existing co-dependent configurations that can span across several administrative boundaries in a distributed, heterogeneous environment.
- 2) To provide efficient handling of the number of potential configurations, that may grow exponentially with each new variability dimension.
- 3) To increase quality and productivity of adaptive system development and help the designers to model, control and validate adaptation policies as well as the trajectory going from one safe configuration to another.

In order to achieve these objectives, we propose a new approach where we combine aspect-oriented and model-driven techniques. The paper is organized as follows. Section 2 describes the DiVA approach. Section 3 introduces the industrial case studies we use to evaluate the approach. Section 4 presents related works. Section 5 provides a conclusion.

2. Proposed Approach

The idea of the approach we propose is to combine model driven and aspect-oriented techniques to handle the complexity of adaptive system construction and execution. Models cope with complexity through abstracting the dynamic variability. Aspect-oriented techniques are utilized to model the adaptation concerns separately from the other aspects of the system. By utilizing model based abstractions and advanced separation of concerns, the adaptation becomes easier to design and understand, possible to validate and allows to easily evolve the adaptation policies even at runtime. In the following sections, we show how we use the two techniques to manage dynamic adaptation.

2.1 Model-driven Approach for Dynamic Adaptation Management

The idea is to use models at two levels in order to manage dynamic variability: at design time and runtime (see Figure 1). In the following we specify how models are used at each level.

At design-time, a system is modelled using a base model which contains its basic functionalities and a set of variant models which can be composed with this base model. In order to identify which variant have to be selected depending on the context of the running system, an adaptation model is specified. The adaptation model specifies the following elements:

- A set of dependencies that specify constraints on variants. For example, the use of a particular variant might require or exclude others. These constraints reduce the total number of configurations by rejecting invalid configurations.
- A context model of the system
- A set of adaptation rules that link the context elements from the context model to the variants that should be used according to the context variation.

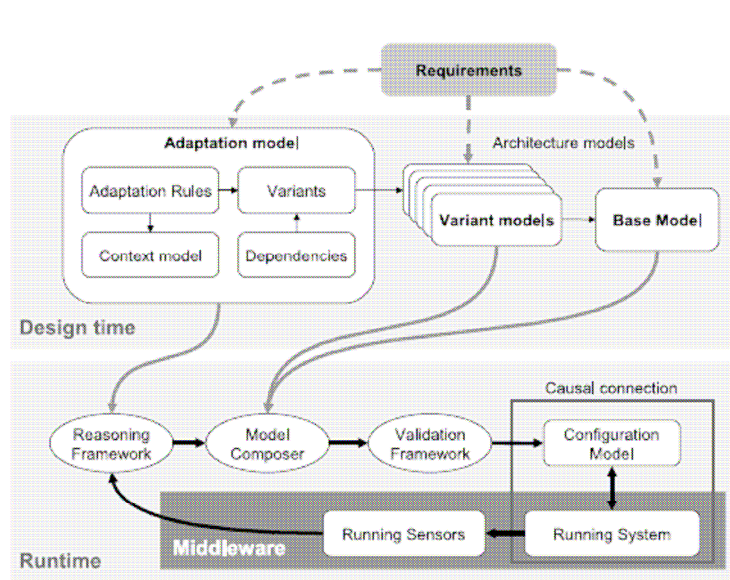


Figure 1. Model-driven Approach for Dynamic Adaptation Management

At runtime the configurations of the system are built by selecting the variants that adapt to the context and composing them with the base. To select the configuration that adapts to the context, the reasoning framework collects context information from sensors and processes the adaptation model. The output of the reasoning framework is one or more options that match the adaptation rules and satisfies the dependency constraints. For each of these options the complete model of the corresponding configuration can be built at runtime using model composition (see more details in Section 2.2). The correctness of the composed configurations are checked with respect to the specified dependencies.

Once a configuration has been selected and checked, the running system can be adapted. The adaptation is carried out through a model that is causally connected to it. This model is transformed to match the configuration that has been selected. The running system is adapted thanks to the causal connection. The main advantage of using causally connected models for dynamic reconfiguration is the automatic generation of reconfiguration scripts instead of writing them by hand.

2.2 Aspect-oriented Modelling Approach for Dynamic Adaptation Management

We use aspect-Oriented Modelling techniques in order to tackle the issue of the combinatorial explosion of variants. AOM allows us to encapsulate distinct variation points into aspects which are separated from the base model of the system's functionality. Then, distinct aspects might be composed into the base model in order to obtain different configurations. This approach allows us to reason on a limited and linearly increasing number of aspects, thus avoiding the specification of all the possible configurations and consequently the problem of combinatorial explosion.

In practice, the variants are defined as aspect models to be woven in the base model. From a particular selection of variants, the corresponding configuration can be built automatically by weaving the corresponding aspect models into the base model.

3. Industrial case studies

Case studies from two different domains are used to validate the DiVA approach:

- The first case study is about the crisis management at an airport where the configuration of an airport systems is dynamically adapted according to the crisis situation. A crisis can cause the failure of one or several critical systems of the airport, it can require a dynamic connection to external systems such a police station, a hospital or a ministry, and it can necessitate the reconfiguration of non-functional services such as the security management and the communication.
- The second case study is about the Customer Relationship Management (CRM) based on dynamic combination of services (service mash-ups) and service adaptation according to the user requirements and preferences.

4. Related Work

Several existing approaches have focused on underlying mechanisms to support dynamic reconfiguration using reflection[1]. Considerable attention has been placed on policies and associated policy languages for reconfigurable systems, and on rule-based approaches and associated interpretation frameworks [3][4]. Significant work focused on autonomic computing leading to the concept of emergent behaviors in evolutionary systems[5]. In DIVA the runtime model uses variation points as a dynamic representation of variability. Thus, the reflection abilities are moved up to the model level. A key benefit is that models can be used to provide a richer semantic base for runtime decision-making related to system adaptation and other runtime concerns. In DiVA we can use models to help determine when a system should move from one consistent architecture to another and compute safe migration paths.

In MADAM and MUSIC[6,7] the main variability mechanism consists in loading different implementations for each component type of the architecture. DIVA reduces the complexity by decomposing the system into a base model and aspects.

5. Conclusion

DIVA proposes a novel combination of Model-Driven Engineering and Aspect-Oriented Modelling to manage dynamic variability of complex systems. Aspect-Oriented Modelling allows to reduce the combinatorial explosion of variants. Model-Driven Engineering automates and improves the creation of the reconfiguration logic. During the next two years, we plan to elaborate our approach by validating it with the industrial case studies and developing a reasoning framework that will automatically select and weave the most adapted aspects. Using a causally connected model to manipulate the running system increases response times for adaptive systems. The evaluation of our approach efficiency is consequently necessary.

6. References

- [1] Coulson, G., Blair, G.S., Clark, M. and Parlavantzas, N. The design of a highly configurable and reconfigurable middleware platform. *ACM Distributed Computing Journal*, 15 (2). 109–126.
- [2] McKinley, P.K., Sadjadi, S.M., Kasten, E.P. and Cheng, B.H.C. Composing Adaptive Software. *IEEE Computer*, 37 (7). 56-64.

- [3] Keeney, J. and Cahill, V., Chisel: A Policy-Driven, Context-Aware, Dynamic Adaptation Framework. in *4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2003)*, (Lake Como, Italy, 2003).
- [4] Capra, L., Emmerich, W. and Mascolo, C., Reflective Middleware Solutions for Context-Aware Applications. in *Third International Conference on Meta-level Architectures and Separation of Crosscutting Concerns*, (Japan, 2001).
- [5] Kephart, J.O. Research challenges of autonomic computing *27th international conference on Software Engineering (ICSE)*, St. Louis, MO, USA, 2005.
- [6] J. Floch, S. Hallsteinsen, E. Stav, F. Eliassen, K. Lund, and E. Gjørven. Using architecture models for runtime adaptability. *Software IEEE*, 23(2):62–70, 2006.
- [7] S. Hallsteinsen, E. Stav, A. Solberg, and J. Floch. Using product line techniques to build adaptive systems. In *SPLC'06: 10th Int. Software Product Line Conf.*, pages 141–150, Washington, DC, USA, 2006. IEEE Computer Society.