Evaluating Competing Dependability Concern Realizations in an Aspect-Oriented Modeling Framework

Robert B. France, Sudipto Ghosh, Indrakshi Ray,
James M. Bieman, Geri Georg, Roger T. Alexander
Computer Science Department, Colorado State University
{*france, ghosh, iray, bieman, georg, rta*}*@cs.colostate.edu*

Abstract

Balancing competing dependability concerns related to security, fault tolerance and performance is a challenge during software development. Addressing these concerns early on in the development lifecycle can reduce the need for extensive and costly design changes later on. We have developed an Aspect-Oriented Modeling framework that allows early localization of competing concerns into aspects and their composition with models reflecting the core functionality. This paper presents approaches for systematic evaluation of the models created and generated in the approach through design analysis and mishap modeling.

Keywords: *Aspect-oriented modeling, dependability, design, dynamic analysis, fault tolerance, mishap modeling, performance, security, static analysis.*

1. Introduction

A secure system restricts access to authorized users and actions. A fault tolerant system continues to operate under adverse conditions. The requirements of secure and fault tolerant systems are interdependent and may conflict. Fault tolerant systems should continue to be available to authorized users while under attack, but they should not be tolerant of attackers. To ensure availability of services through replication, developers may add functionality that ensures the confidentiality, integrity and currency of replicated data, but impacts performance.

System implementations, in general, tend to have program code to satisfy these competing concerns mixed up with code that provides code functionality. Such distributed code is likely to be difficult to maintain. We have developed an Aspect-Oriented Modeling (AOM) approach that models these competing concerns at the design level, so that design solutions can be developed, analyzed, compared and improved. Our AOM approach localizes cross-cutting realizations of dependability concerns in aspect models using the Unified Modeling Language (UML) diagram template notation [3, 4]. The primary model describes realizations of core functionality in the form of UML design models. The integrated design model consists of an integrated set of aspect models and primary models. Developers compose the aspect and primary models using mapping rules and composition directives to produce a full design model of the system. Details of the approach can be found in [3, 4].

All the models produced in this approach — aspects, primary and composed — can be subjected to different types of analysis to ensure that both core functionality and dependability features are preserved. The order of composition may make a difference in the resulting model properties. Conflicts may occur during the composition process for various reasons. A conflict occurs when operations with the same signature but different behavioral specifications (i.e., pre- and post-conditions) appear in the models. A behavior required by an aspect may not be performed as specified because its sub-behaviors may have been modified after behaviors are merged with other aspect models. A conflict may also occur when a system needs to compose aspects with seemingly inconsistent functionality: for example, a security aspect and a fault tolerance aspect. The security aspect tries to reduce availability, but the fault tolerance aspect tries to increase it.

2. Analysis of Aspect-Oriented Models

Analysis of the models is performed to identify design deficiencies. A deficiency can be a stress point, which is a non-robust part of a design that can be exploited to compromise security or other desired system behavior, a conflict, or a design flaw.

The identification of design deficiencies and assessment of associated risks involves mishap modeling [2], in which we anticipate and model misuses of the system. Consider a system that requires password-based authentication of users

accessing mission-critical services. This system also requires duplication of mission critical services on geographically remote computers on a wide area network (WAN). A system that satisfies the requirements may allow users to access mission-critical services from remote locations after authenticating them with passwords. However, if the passwords are not encrypted when transmitted on the WAN, the communication of the passwords becomes a design stress point. Design stress points can also lead to denial or degradation of services. In these cases, the stress arises when transaction loads or other performance related loads rise above or fall below certain thresholds.

In our approach, we model anticipated situations that *stress* or misuse the system. UML diagrams are used to describe misuse scenarios. For example, to evaluate the impact of security concerns on an application, we define test scenarios that represent malicious attacks and sanctioned interactions. These scenarios act as test data that determine if the design elements described by security aspects are sufficient to prevent the attacks from compromising protected resources. Sanctioned interaction scenarios can help determine the impact security aspects have on authorized activities. Test scenarios are expressed in terms of UML behavioral models (e.g., sequence diagrams) and can be based on use cases that describe authorized behaviors and misuse cases [2] that describe behaviors that should not be present in a correct system implementation.

The test scenarios can be used for both static and dynamic analysis of the models. The selection and quality of selected scenarios helps to determine the effectiveness of scenario-based testing of the composed models.

Three levels of analysis can be used on the models:

1. *Unit* analysis occurs when a single aspect model is analyzed in isolation.
2. *Integration* analysis occurs on the model resulting from composing an aspect with the primary model.
3. *System* analysis occurs when all the aspects have been composed with the primary model.

When multiple aspects are composed sequentially with a primary model, one must check that no existing capabilities were broken and that the capability of the newly composed aspect was preserved in the composition. This entails regression analysis and also analyzing the newly composed aspect.

In the static approach, we compose the mishap models with the integrated aspect-oriented design model. If the composition results in a consistent model, it implies that the design allows the compromised behaviors described by the misuse scenarios. Another static analysis approach uses Paltor's model-checking technique [5].

In the dynamic analysis approach, we *test* the models using a formally defined operational semantics for UML models [1]. The UML diagrams specify a system of asynchronously communicating objects whose behaviors are described by state machines. The misuse scenarios are used as test inputs in this analysis.

3. Conclusions and Future Work

Several techniques can be used for analyzing models in our AOM framework. Model analysis can lead to the identification of design deficiencies and assessment of risk associated with the stress points. Developers may perform systematic trade-off analysis on a set of alternative realizations of dependability concerns to select the *best* option that meets design objectives and minimizes risks.

We are working on developing prototype tools that can assist in the analysis of models. We are also evaluating the effectiveness of our approach using various dependability concerns.

References

[1] A. Andrews, R. France, S. Ghosh, and G. Craig. Test Adequacy Criteria for UML Design Models. *Journal of Software Testing, Verification and Reliability*, 13(2):95-127, April-June, 2003.

[2] I. Alexander. Misuse Cases: Use Cases with Hostile Intent. *IEEE Software*, 20(1):58–66, January-February, 2003.

[3] G. Georg, R. B. France and I. Ray. Using Aspects to Design a Secure System. In *Proceedings of the International Conference on Engineering Complex Computer Systems (ICECCS 2002)*, ACM press, Greenbelt, MD, December, 2002.

[4] G. Georg, I. Ray and R. B. France. Designing High Integrity Systems Using Aspects. In *Proceedings of the Fifth IFIP TC-11 WG11.5 Working Conference on Integrity and Internal Control in Information Systems (IICIS 2002)*, Bonn, Germany, November, 2002.

[5] I. P. Paltor. Modeling and Analyzing Softrware Behavior in UML. Ph.D. thesis, Turku Center for Computer Science.