Semantic-Based Aspect Interaction Detection with Goal Models (position paper)

Gunter MUSSBACHER [a,1], Daniel AMYOT [a], and Jon WHITTLE [b]
[a] *SITE, University of Ottawa, Canada*
[b] *Department of Computing, InfoLab21, Lancaster University, UK*

Abstract. Detecting undesired interactions in aspect-oriented models is closely related to detecting undesired feature interactions. Aspect interactions can be broadly categorized into syntactic interactions, which can be discovered by analysis based on syntax, and semantic interactions, which require an interpretation of the meaning of models. Semantic interactions are very hard to detect and at the same time often require significant rethinking or remodeling of the aspects. We argue that more research is needed to effectively deal with semantic interactions in aspectual models and that goal models are a suitable candidate to reason about these interactions.

Keywords. Aspect-Oriented Modeling, Semantic-Based Feature Interactions, Goal Models, Scenario Models, User Requirements Notation

1. Introduction

Although feature interactions (FIs) are most frequently researched in the field of telecommunications [1][2], they are common to a wide range of domains and applications. For example, in the International Conference of Feature Interactions (ICFI) series, feature interactions have been discussed in the context of control systems and web services [3], policies and embedded systems [2], software product lines and enterprise information systems [4], as well as business processes and licensing [5] . With the emergence of aspect-orientation as a modeling technique to structure software artifacts of complex systems, FIs have to be detected and managed in yet another context, i.e., in the context of Aspect-Oriented Modeling (AOM), which concerns itself with the specification, composition, and analysis of aspects during requirements and design. In this paper, we focus on FIs in the context of aspect-oriented requirements models and employ the Aspect-oriented User Requirements Notation (AoURN) [6][7] as our modeling language.

Aspect interactions may occur when multiple aspects are applied to the same location in the base model. Straightforward ordering of aspects may resolve simple interactions but complex cases with deep semantic conflicts between aspects may require a rethinking of which aspects to apply or a remodeling of the aspects themselves. Further-

[1] Corresponding Author: SITE, University of Ottawa, 800 King Edward, Ottawa, ON, K1N 6N5, Canada; E-mail: gunterm@site.uottawa.ca.

more, whether semantic conflicts exist between aspects depends on the system under consideration and the evolving needs of stakeholders, as explained below.

Syntactic interactions can be detected by comparing syntax. A simple example is when one aspect depends on certain elements that are only introduced by another aspect – an ordering of the aspects easily resolves this interaction. In previous work, we applied critical pair analysis to detect syntactic interactions between aspect models [8]. Although useful in practice, this technique is limited in that it cannot handle a whole class of other interactions, which we call *semantic interactions*. An example of a complex semantic interaction is a conflict between security and performance aspects. Security inevitably impacts performance because of additional processing and delays. Performance may also impact security, if the performance aspect caches results, which must then be protected. However, the conflict between security and performance is only relevant if both are important to the stakeholders. If, for example, security is not an issue for the stakeholders, then the performance aspect may be applied unrestrictedly. This situation is further complicated by the differing and possibly conflicting needs of various stakeholders of the system which have to be considered and balanced. Semantic interactions, therefore, require a context-based interpretation of the meaning of models – e.g., those models related to performance and security.

In this paper, we propose a novel research direction based on goal models and semantic annotations for the detection of a subclass of semantic interactions and present the first initial steps in that direction. We augment aspect models with lightweight *semantic annotations* and model the semantic impact of aspects on each other in a goal model called an *influence model* [9]. When multiple aspects are applied at the same point, the influence model can be used to identify and trade-off semantic aspect interactions. Goal models are ideally suited for this purpose since they are good at capturing qualitative relationships [6]. Goal models enable us also to capture the needs of various stakeholders, the alternatives (i.e., aspects) that are considered to address these needs, and the impact of these alternatives on all stakeholders. The goal model can then be used to simultaneously reason about stakeholder needs and aspect interactions with the help of qualitative or quantitative evaluation mechanisms [6] to find the most appropriate trade-off for all stakeholders.

The remainder of this paper describes the proposed framework, presents related work and conclusions, and discusses future research directions.

2. Framework for Semantic-Based Aspect Interaction with Goal Models

Our proposed framework is based on a lightweight semantic interpretation of model elements and relies on a set of *semantic markers* (domain-specific annotations for each aspect domain) and an *influence model* (a goal model that shows how annotations from different domains influence each other). A prototype of the proposed framework has been implemented with the Aspect-oriented User Requirements Notation (AoURN) [6][7] which combines goal (GRL – Goal-oriented Requirement Language), scenario (UCM – Use Case Maps), and aspect modeling in one language. Base and aspect models are defined with individual UCM scenario models and composed into new UCM scenario models, semantic markers are implemented as domain-specific URN metadata and can be applied to any model element, the influence model is a GRL goal model, and the analysis of the goal model uses existing evaluation mechanisms offered by URN.

Goal models allow us to use well-defined relationships for specifying negative or positive influences while leveraging existing analysis techniques to automatically analyze interactions between aspects from different domains and to reason about trade-offs between conflicting aspects. The GRL model in Figure 1 depicts an example influence model with four aspect domains (and five semantic markers): distribution (<<remote>>, <<local>>), authentication (<<confidential>>), performance (<<cache>>), and encryption (<<encrypted>>). Note how the influence model shows that Authentication improves Confidentiality while Remote Server hurts it.

Two types of nodes have been used in this GRL model. Aspects are modeled with tasks (⬡, e.g., Authentication) while the non-functional requirements associated with the aspects are modeled with softgoals (⬭, e.g., Confidentiality). Softgoals describe something to be achieved that cannot be measured quantitatively but is of a qualitative nature. This does not mean that the softgoal itself cannot be measured (e.g., performance can certainly be measured with metrics such as throughput, response time, etc.), but the question is rather how much is enough (e.g., how much performance is enough to have achieved what needs to be achieved). Tasks, on the other hand, describe possible solutions to achieve the softgoals. Contribution links (→) indicate the impact of GRL nodes on each other. Correlations (⇢) are similar to contributions in that they also indicate impact but are used to describe side effects rather than desired impacts. For this GRL model, the impact of an aspect on its own softgoal is shown as a contribution (e.g., Caching on Performance) and the impact of an aspect on softgoals of other aspects is shown as a correlation (e.g., Caching on Confidentiality). Negative and positive influences are indicated by the labels of the links. Positive influences may be sufficient ($\dot{+}$), insufficient ($\underset{\cdot}{+}$) or some positive (+). Similar levels exist for negative influences (−).

Hence, the links in the goal model connect abstract, high-level goals captured during the early requirements phase with aspect-oriented scenario models from the requirements and early design phases. Each task in the goal model is modeled in greater detail with UCMs.

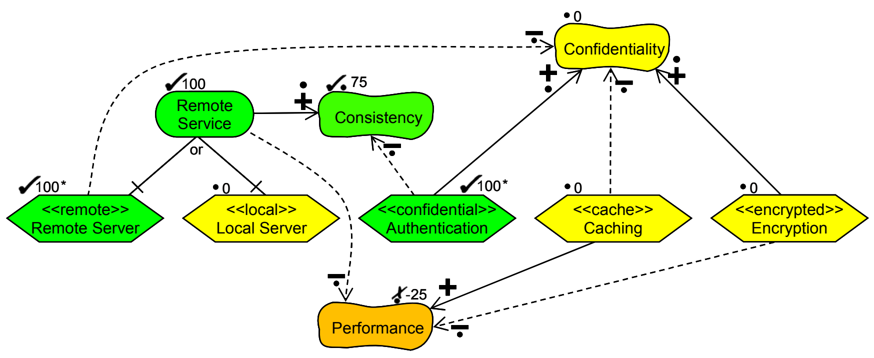

Figure 1. Influence Model

The UCM model in Figure 2 shows a simple Reporting use case with three responsibilities (✕, e.g., presentOptions) assigned to three components (□, e.g., Voting Machine). The use case is described by a path which starts at a start point (●, e.g., report) and ends at an end point (▌, e.g., reported).

Two aspects and the composed model where both aspects have been applied are also shown. The pointcut stub (P, e.g., requiresAuthentication) is a placeholder for the actual pointcut expressions of the aspect. A pointcut expression is a pattern that must be matched in the base model if the aspect is to be applied, thus determining the base model locations to which the aspect is applied. The causal relationship of the pointcut stub with the rest of the UCM defines the composition rule for the aspect (e.g., since the behavior described for the Authentication aspect occurs before the pointcut requiresAuthentication, the aspectual behavior is added before the matched location in the base model).

The actual pointcut expressions for the two pointcut stubs are shown in Figure 3. Note that pointcut expressions are at the same abstraction level as the scenario models used to describe the aspects. Grey, unnamed start and end points are not included in the pattern but only denote the beginning and end of the pointcut expression. Therefore, the Authentication aspect matches against the presentOptions responsibility in the Reporting use case and adds authentication behavior which may fail before this responsibility. The Remote Service aspect matches against the authenticate responsibility in the Authentication aspect and moves it from the Local Server to the Remote Server. Note that the Remote Service aspect is modeled in a very application-specific way only to simplify the example.

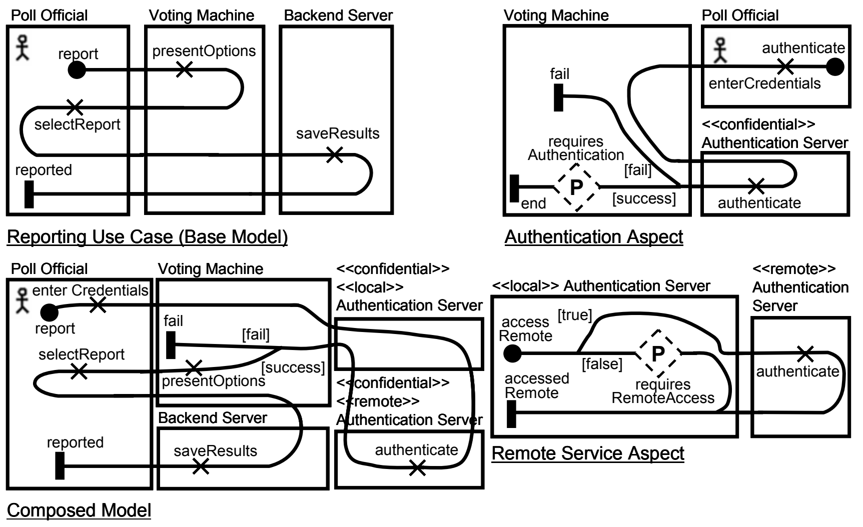

Figure 2. Base Model, Aspect Models, and Composed Model

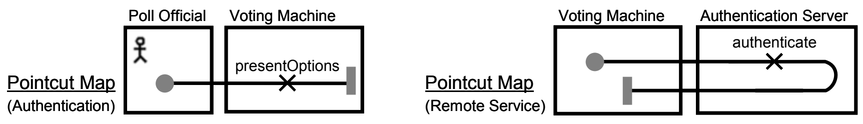

Figure 3. Pointcut Expressions for the Authentication and Remote Service Aspects

The framework consists of a 4-step process. In step 1, each individual aspect model is manually annotated with *semantic markers* that define an interpretation of relevant model elements. For example, in the Authentication aspect model in Figure 2, a server that stores sensitive data is annotated with <<confidential>>. In step 2, the desired aspects are applied to a model. The aspect composition mechanism yields a new model where some model elements are now annotated with semantic markers from several aspects. For example, the server is now annotated with both <<confidential>> and <<remote>> as the Authentication and Remote Service aspects have both been applied to the model.

In step 3, the influence model is instantiated based on the existence of semantic markers for a model element in the composed model. An influence model is prepared for evaluation for each model element with more than one semantic marker. For example, the Authentication Server in the composed model is annotated with <<remote>> and <<confidential>>. Therefore, the evaluation values of the corresponding model elements in Figure 1 are set to 100 (the * indicates these initial evaluation values). All other model elements default to the evaluation value 0 on the GRL scale of [-100, 100].

Finally, step 4 analyzes the influence model for potential conflicts between semantic markers by propagating the initial evaluation values to the top level elements in the goal tree with the help of existing evaluation mechanisms. For example, the top goal Confidentiality in Figure 1 does not have a high evaluation value, and is therefore flagged as a possible conflict if the goal is important to the stakeholder. This accurately reflects that sending confidential data from a remote server across a network is problematic and security is an issue for the stakeholder. With this in mind, the modeler is now in a position to address the aspect interaction by updating the aspects, the influence model, or semantic markers as required and re-iterating through the process.

For more details about the framework, the reader is referred to [9].

3. Related Work

Blair and Pang [10] proposed a two-level architecture that cleanly separates a feature's core behavior (in Java) from aspect-oriented code (in AspectJ), specifying how features interactions are resolved. They illustrated their approach with an email case study. However, they have not addressed the aspect interaction problem, which is still largely unsolved. One approach has been to explicitly document interactions such as aspect interaction templates [11], precedences [11][12], and aspect interaction charts [13]. This does not offer automated help in detecting interactions and does not take the changing needs of stakeholders into account. Formal methods (e.g., model checking [14], formal specifications of pre- and post-condition [15], or static analysis [16][17]) have been applied to detect interactions, but are effort-intensive.

We are not aware of any work that takes into account the semantics of model elements when detecting interactions with the help of lightweight semantic annotations and goal models. Rather, FI approaches are typically based on detecting structural interactions (e.g., [18]) or on applying formal methods such as model checking [19]. The idea of semantically-informed aspect development, however, builds upon previous work in semantic-based aspect composition for natural language requirements documents [20] and scenarios [21]. Detection of semantic interaction was also discussed for aspect-oriented programming (AOP) [22].

Goal models, however, have been used in FI research. Metzger *et al.* [23] use goal models in the context of software product lines to describe overall system goals that are decomposed into a set of features. A simple static analysis of the goal model then identifies points of interaction. The goal model, however, does not take several stakeholders into account, only models decomposition of features but no positive or negative impacts between features, and therefore does not reason about these relationships. Weiss *et al.* [24] use goal and scenario models in the context of web service FIs. Several stakeholders are considered in the goal model, the impacts of features on each other are considered, and the goal model is analyzed with an evaluation mechanism that takes these relationships into account. As this is not an aspect-oriented approach, the aspect composition mechanism is not utilized to identify candidates for possible FIs.

4. Conclusion and Future Work

We proposed a new research direction for semantic-based aspect interaction detection with lightweight semantic annotations and goal models. A promising, initial, proof-of-concept evaluation of the approach has been performed in [9], but further experiments with larger models and eventually with standard profiles such as MARTE [25] need to be carried out. We expect reusable, application-independent aspect models to be developed incrementally and annotated with semantic markers. Similarly, influence models need only be defined once and updated with new aspects. More challenging work is required to develop widely accepted sets of semantic markers for each aspect domain, ideally based on domain-specific languages and profiles, and to capture their positive and negative impacts on each other in influence models.

The example in this paper presents semantic markers that are all at the same abstraction level. However, semantic markers will likely be structured hierarchically, i.e., one general security marker may be broken down into several lower-level semantic markers that deal with security specifics. In such a case, the influence model can also be structured hierarchically to reflect the organization of the semantic markers and to enable reasoning about semantic markers that span various abstraction levels.

The framework also needs to be extended to clearly define the relationship between stakeholder goal models and influence models. This is complicated by the fact that for large systems, the resolution of the same semantic aspect interaction may be different as the importance of high-level goals may change for stakeholders from one situation to the next. These variations need to be captured in the goal models and adequately considered when reasoning about trade-offs for a particular interaction. Finally, the preferred of several evaluation mechanisms needs to be found through empirical evaluations, and the automation of the process should be improved.

References

[1] Bouma, L.G., Griffeth, N., and Kimbler, N. (Editors), "Feature Interactions in Telecommunications Systems", *Computer Networks* ***32:4*** (2000).

[2] Amyot, D. and Logrippo, L. (Editors), "Directions in Feature Interaction Research", *Computer Networks* ***45:5*** (2004).

[3] Amyot, D. and Logrippo, L. (Editors), *Feature Interactions in Telecommunications and Software Systems VII*, IOS Press, 2003.

[4] Reiff-Marganiec, S. and Ryan, M.D. (Editors), *Feature Interactions in Telecommunications and Software Systems VIII (ICFI)*, IOS Press, 2005.

[5] du Bousquet, L. and Richier, J.-L. (Editors), *Feature Interactions in Software and Communication Systems IX (ICFI)*, IOS Press, 2008.

[6] ITU-T: *Recommendation Z.151 (11/08): User Requirements Notation (URN) – Language definition*, Geneva, Switzerland, 2008.

[7] Mussbacher, G., "Aspect-Oriented User Requirements Notation", Giese, H. (Editor), *Models in Software Engineering: Workshops and Symposia at MODELS 2007*, Springer, LNCS 5002 (2008), 305-316.

[8] Jayaraman, P., Whittle, J., Elkhodary, A., and Gomaa, H., "Model Composition in Product Lines and Feature Interaction Detection using Critical Pair Analysis", *MODELS 2007*, Nashville, USA. LNCS 4735 (2007), Springer, 151-165.

[9] Mussbacher, G., Whittle, J., and Amyot, D., "Towards Semantic-Based Aspect Interaction Detection", *NFPinDSML 2008*, workshop at *MODELS 2008,* Toulouse, France, 2008.

[10] Blair, L. and Pang, J. "Aspect-Oriented Solutions to Feature Interaction Concerns using AspectJ", *Feature Interactions in Telecommunications and Software Systems VII*, IOS Press, 2003, 87-104.

[11] Sanen, F., Loughran, N., Rashid, A., Nedos, A., Jackson, A., Clarke, S., Truyen, E., and Joosen, W., "Classifying and Documenting Aspect Interactions", *Workshop on Aspects, Components and Patterns for Infrastructure Software at AOSD*, Bonn, Germany, 2006, 23-26.

[12] Zhang, J., Cottenier, T., Van den Berg, A., and Gray, J., "Aspect Composition in the Motorola Aspect-Oriented Modeling Weaver", *Journal of Object Technology* ***6*** (2007), 89-108.

[13] Bakre, S. and Elrad, T., "Scenario based resolution of aspect interactions with aspect interaction charts", *10th Intl. Workshop on Aspect Oriented Modeling (at AOSD)*, Vancouver, Canada, 2007, 1-6.

[14] Shaker, P. and Peters, D., "Design-Level Detection of Interactions in Aspect-Oriented Systems", in *Aspects, Dependencies and Interactions Workshop at ECOOP 2006*, 2006.

[15] Mostefaoui, F. and Vachon, J., "Design-level Detection of Interactions in Aspect-UML models using Alloy", *Journal of Object Technology* ***6*** (2007), 137-165.

[16] Douence, R., Fradet, P., and Südholt, M., "Composition, Reuse and Interaction Analysis of Stateful Aspects", *Aspect Oriented Software Development*, 2004, 141-150.

[17] Douence, R., Fritz, T., Loriant, N., Menaud, J.-M., Segura-Devillechaise, M., and Südholt, M., "An Expressive Aspect Language for System Applications with Arachne", *Aspect-Oriented Software Development (AOSD)*, Chicago, Illinois, 2005, ACM, 27-38.

[18] Liu, J., Batory, D., and Nedunuri, S., "Modeling interactions in feature oriented systems", in *Feature Interactions in Telecommunications and Software Systems VIII (ICFI)*, 2005, IOS Press, 178-197.

[19] du Bousquet, L., "Feature Interaction Detection using Testing and Model Checking: Experience Report", in *World Congress on Formal Methods in the Development of Computing Systems*, 1999. LNCS 1708, Springer, 622-641.

[20] Chitchyan, R., Rashid, A., Rayson, P., and Waters, R., "Semantics-Based Composition for Aspect-Oriented Requirements Engineering", *AOSD 2007*, Vancouver, Canada, 2007, ACM, 36-48.

[21] Klein, J., Hélouët, L., and Jézéquel, J.-M., "Semantic-Based Weaving of Scenarios", *Aspect-Oriented Software Development (AOSD)*, Vancouver, Canada, 2006, ACM, 27-38.

[22] Bergmans, L.M.J., "Towards Detection of Semantic Conflicts between Crosscutting Concerns", *Analysis of Aspect-Oriented Software (AAOS), workshop at ECOOP 2003*, Darmstadt, Germany, 2003.

[23] Metzger, A., Bühne, S., Lauenroth, K., and Pohl, K., "Considering Feature Interactions in Product Lines: Towards the Automatic Derivation of Dependencies between Product Variants", *Feature Interactions in Telecommunications and Software Systems VIII (ICFI)*, 2005, IOS Press, 198-216.

[24] Weiss, M., Esfandiari, B., and Luo, Y., "Towards a classification of web service feature interactions", *Computer Networks* ***51:2*** (2007), 359–381.

[25] OMG, *UML Profile for Modeling and Analysis of Real-time and Embedded Systems (MARTE)*, 1.0 Beta 2 Specification (ptc/08-06-09), June 2008.