

On Modeling Interactions of Early Aspects with Goals

Gunter Mussbacher and Daniel Amyot
SITE, University of Ottawa
800 King Edward
Ottawa, ON, K1N 6N5, Canada
{gunterm | damyot}@site.uottawa.ca

Abstract

Interactions in aspect-oriented models must be detected, documented, and resolved for aspects to be composed as desired. Generally, aspect interactions can be categorized as intrinsic (those that inherently exist among concerns) or technical (those that are dependent on technology and may change over time). Consequently, these types of interactions should be encapsulated properly. Goal models support reasoning about qualitative and quantitative relationships and are therefore ideally positioned to describe and reason about intrinsic interactions, because they are often of a qualitative nature. On the other hand, technical interactions are typically syntactic conflicts and dependencies which are modeled with different techniques. We present the Concern Interaction Graph (CIG), a goal model specialized for technical interactions in aspect-oriented models, which is integrated with other goal models for intrinsic concern interactions and stakeholder intentions. The CIG therefore allows global trade-offs among concerns that take intrinsic and technical interactions into account as well as the needs of stakeholders, while maintaining proper separation of concerns between intrinsic and technical interactions.

1. Introduction

A significant contribution of aspect-oriented modeling is the focus on the formalization of and reasoning about relationships between concerns. Composition rules formally describe to a certain extent these relationships. At the early stages of software development, however, concerns strongly overlap with broadly scoped qualities such as performance, reliability, and security with relationships that are often of a qualitative nature. These relationships go well beyond com-

mon composition rules such as before, after, around, concurrent, and interleaved [1].

For example, a security concern surely impacts negatively a performance concern, because more resources are required for security features such as encryption, authentication, or access control, but just how much? On the other hand, a performance concern can affect negatively a security concern if the performance concern caches results, which must then be protected. These are examples of qualitative interactions. Interactions are one of the most interesting kinds of relationships between concerns, because they describe potentially undesirable impact of one concern on another. Interactions have been studied extensively in the telecommunications domain [2] but are applicable to many other domains and applications. Interactions manifest themselves in aspect-oriented models, when multiple aspects affect the same elements in the base.

Many interactions may be categorized as either *intrinsic* or *technical*. The above-mentioned security-performance interactions are an example of intrinsic interactions in addition to being qualitative, because, regardless of the technologies used to compose these concerns, security will impact performance. These interactions are often very hard to detect, and in complex cases, deep intrinsic interactions between concerns may require a) a rethinking of which concerns should be applied or b) a remodeling of the concerns themselves. Moreover, the significance of these interactions also depends on the importance of the concerns to the system's stakeholders. For example, if a stakeholder is not concerned about security at all, a caching mechanism can be applied without restraint.

Goal models have long been used for the description of such qualitative relationships, i.e., interactions that are hard to describe with quantitative methods, as well as for the description of stakeholder intentions [3]. In previous work, we have applied goal models to intrinsic interactions in aspect-oriented models

(called semantic interactions at that time) [4], which allows reasoning about this type of interactions in the context of the goals of stakeholders.

Technical interactions, on the other hand, cover dependencies and conflicts between concerns that arise because of the way concerns are built and composed. They are dependent on current technology and therefore subject to change as opposed to intrinsic interactions that are a fundamental phenomenon of the involved concerns. Technical interactions are often syntactic in nature, and can often a) be detected by comparing syntax and b) be resolved by an ordering that respects their dependencies and conflicts. An example of a dependency is a concern that assumes certain elements in the base model that are only introduced by another concern. Conflicts, on the other hand, describe any other situation where the ordering of concerns matters. These interactions can typically be quantified and therefore have been modeled with different techniques [5][6][7][8][9] other than goal models.

We argue that technical interactions should also be modeled with goals to enable simultaneous reasoning about both kinds of concern relationships: intrinsic and technical. However, both types of interactions should remain separate as technical interactions are bound to change as technologies evolve. We therefore introduce the *Concern Interaction Graph* (CIG) which has been developed for the Aspect-oriented User Requirements Notation (AoURN) [10], a framework that combines goal-oriented, scenario-based, and aspect-oriented modeling in one notation for requirements engineering. A CIG is a new view of AoURN goal models that focuses on technical concern interactions. The CIG is seamlessly integrated with other AoURN goal views that describe in a qualitative way a) intrinsic concern interactions and b) the intentions and relationships of stakeholders. With a CIG, constraints imposed by technical interactions may now be taken into account in a global analysis of trade-offs among different concerns.

Furthermore, given dependencies and conflicts, the CIG defines an ordering among concerns that resolves these interactions or indicates those that cannot be or have not yet been resolved. The ordering can then be used when concerns are composed. Moreover, the order may be influenced by the results of the global analysis of trade-offs.

This research is an extension of our previous work on modeling semantic interactions [4]. While our previous work supported the detection of possible semantic interactions, the scope of a CIG is not the detection of technical interactions but rather the documentation of their existence and their resolution. Existing detection techniques may be used to populate the CIG.

In the remainder of this paper, section 2 provides a brief overview of AoURN and related work. Section 3 presents a sample AoURN model that illustrates intrinsic interactions and will be extended with a CIG in section 4. This is followed by our conclusion and discussion of future work in section 5.

2. Background

2.1. Aspect-oriented URN

The User Requirements Notation (URN) is a recent International Telecommunication Union (ITU) standard for capturing early requirements [3]. URN consists of two complementary sub-languages called Goal-oriented Requirement Language (GRL) and Use Case Maps (UCM) for goal-oriented and scenario-based modeling, respectively. GRL models are used to describe and reason about non-functional requirements (NFRs), quality attributes, and the intentions of system stakeholders, while UCM models are used for operational requirements, functional requirements, and performance and architectural reasoning. In summary, URN has concepts for the specification of stakeholders, goals, NFRs, rationales, behaviour, actors, scenarios, and structuring. The subset of the GRL notation (Fig. 1) relevant to this paper will be explained in more detail for the example in section 3.

The Aspect-oriented User Requirements Notation (AoURN) is a modeling framework that extends URN with aspect-oriented concepts [10], allowing modelers to better encapsulate crosscutting concerns which are hard or impossible to encapsulate only with URN models. AoURN treats concerns as first-class modeling elements, regardless of whether they are crosscutting or not. Typical concerns in the context of URN are a stakeholder's intentions, NFRs, and use cases. AoURN adds aspect concepts to URN's sub-languages, leading to and integrating Aspect-oriented GRL (AoGRL) [11] and Aspect-oriented UCMs (AoUCM) [12][13]. AoURN groups all relevant properties of a concern such as goals, behavior, and structure, as well as point-

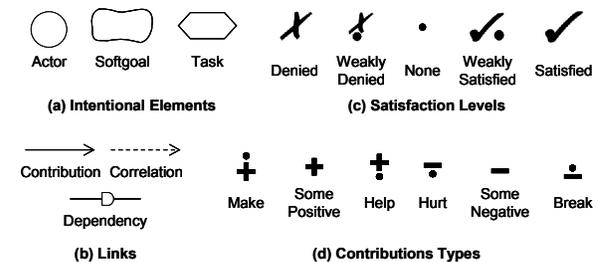


Fig. 1. Subset of GRL notation

cut expressions needed to apply new goal and scenario elements to a base model or to modify existing elements. A *pointcut expression* is a pattern that must be matched in the base if the concern is to be applied, thus determining the base elements to which the concern is applied. AoURN uses standard URN diagrams to describe pointcut expressions (i.e., it is only limited by the expressive power of URN itself as opposed to a particular composition language). Finally, AoURN employs an aspect composition technique that can fully transform URN models.

While the URN metamodel already defines the concept of a concern including a condition that can be used to enable the concern, it does not capture explicitly concern dependencies and conflicts, nor is there any concrete syntax to visualize these relationships.

2.2. Related Work

There have been few attempts to handle aspect interactions during modeling. These approaches can be grouped into those that document interactions and those that detect interactions. Aspect interaction templates [8], precedences [8][9], and aspect interaction charts [5] explicitly document interactions. Kienzle et al. [7] define inter-aspect dependencies but do not consider conflicts. Formal methods (e.g., model checking [14] or static analysis [15]) have been applied to detect technical interactions, but are effort-intensive. Critical pair analysis as employed by MATA [6] works well for technical interactions but cannot handle intrinsic interactions. It uses a numeric ordering scheme to capture precedence.

None of these approaches are integrated with models that describe stakeholder intentions or the intrinsic concerns interactions. Hence, they are not suitable to reason about the overall system and the global trade-offs between concerns and stakeholder intentions.

In previous work, we captured intrinsic interactions (called semantic interactions at that time) [4] to

reason about inherent interactions between concerns based on the semantics of concern elements with the help of goal models and alert the modeler to possible interactions. We are not aware of any other work that goes beyond technical interactions and takes into account deep semantics of model elements when detecting interactions. The idea of semantically-informed aspect development, however, builds upon previous work in semantic-based aspect weaving. For example, in aspect-oriented requirements engineering, Chitchyan et al. [16] use natural language processing to take into account English semantics when composing textual documents. For modeling, Klein et al. [17] weave UML sequence diagrams by matching semantically equivalent but syntactically different sequences.

Goal models, however, have been used for the feature interaction problem [18][19], but not in the context of aspect-oriented software development. Metzger et al. [20] use goal models in the context of software product lines to describe overall system goals that are decomposed into a set of features. Stakeholders as well as positive and negative impacts between features are not taken into account. Weiss et al. [21] use goal and scenario models in the context of web service feature interactions. As these are not aspect-oriented approaches, the differences between technical and intrinsic aspect interactions are not taken into account.

3. Example

The standard GRL goal model in Fig. 2 illustrates the intrinsic interactions between four sample concerns: Remote Service, Authentication, Caching, and Encryption. A GRL goal model connects *intentional elements* with contribution and correlation links. Elements are called intentional because they carry stakeholder intentions. Two types of intentional elements have been used in this model. Softgoals (\square , e.g., Confidentiality) describe something to be achieved that cannot be measured quantitatively but is of a qualitative nature, while tasks (\diamond , e.g., Authentication) in-

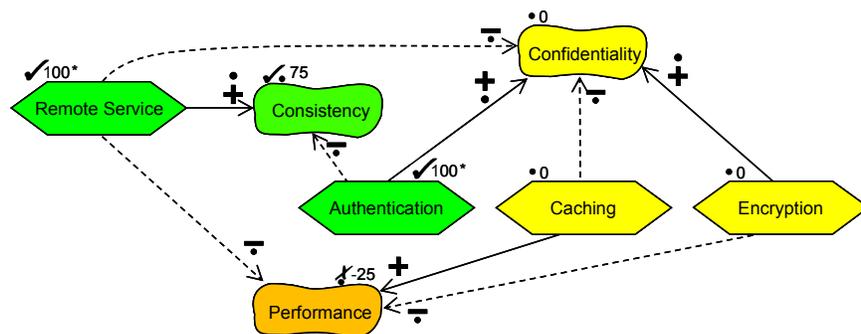


Fig. 2. Intrinsic interactions

dicating possible solutions. In this case, concerns are modeled with tasks while the NFRs associated with the concerns are modeled with softgoals. Capabilities defined as GRL tasks may further be refined in UCM models, which include richer concepts for scenarios.

Contribution links (\rightarrow) indicate the impact of intentional elements on each other. Correlations (\dashrightarrow) are similar to contribution links in that they also indicate impact but are used to describe side effects rather than desired impacts. For this goal graph, the impact of a concern on its own softgoal is shown as a contribution (e.g., Caching on Performance) and the interactions of a concern with softgoals of other concerns are shown as correlations (e.g., Caching on Confidentiality). The labels on the arrows represent qualitative *contribution types* (Fig. 1.d) and indicate the levels of positive or negative impact.

In this simple example, each concern is modeled by a task and its softgoal – Confidentiality for Authentication, Consistency for Remote Service, Performance for Caching, and finally Confidentiality for Encryption. The links and contribution types indicate that Authentication has a positive impact on Confidentiality (contribution type: Help). Remote Service as well as Caching, however, negatively impact Confidentiality (Hurt), because data transferred across a network as well as cached data is potentially vulnerable to security attacks. Encryption, on the other hand, ensures Confidentiality is achieved (Make). Caching improves performance (SomePositive) and Authentication does not have considerable negative or positive performance implications and is therefore neutral (no link), but both Encryption and Remote Service result in significant performance penalties (Hurt) because of additional processing and network delays, respectively. In terms of Consistency, Authentication as a non-remote service is problematic (Hurt), because the distribution and update of data to local machines needs to be managed at setup time. A Remote Service, however, ensures that Consistency is achieved as the most-up-to-date information is always accessed (Make).

Note how these interactions are rather independent from the actual application domain and can be reused in many contexts.

For evaluation purposes, initial *satisfaction values* can be associated with any intentional element and are indicated by a * in Fig. 2. In the case of the GRL goal model for intrinsic interactions, the satisfaction values of the selected concerns are initialized and then propagated by the GRL evaluation mechanism to satisfaction values of high-level goals, thus providing an assessment of the suitability of the proposed group of concerns. This is supported by the *jUCMNav* tool [22], the most comprehensive URN tool. On the GRL scale

of [-100, 100], the satisfaction value (Fig. 1.c) of a chosen concern is typically set to the maximum quantitative value 100 (and its corresponding highest qualitative value Satisfied), while all other concerns are set by default to 0 (None). Fig. 2 shows the impact on the high-level goals if Remote Service and Authentication are applied: Consistency is reasonably satisfied (value 75) whereas Confidentiality is only partially satisfied (0), and Performance may be problematic (-25). This is not a desirable result and an indication to the modeler to rethink the current selection of concerns (e.g., by adding Caching to improve Performance).

4. Concern Interaction Graph

As the focus of this paper is on intrinsic and technical concern interactions and given the space constraints, the actual AoURN models of the example domain, an application submission system, are not shown. It is sufficient to know that there are two stakeholder concerns, the Applicant and the System Provider, a Submit Application use case concern with technical dependencies on the Applicant concern, and the four NFR concerns from Fig. 2. Authentication is applied to the use case concern and to the System Provider concern, Remote Service is applied to Authentication as well as the use case concern, and Caching and Encryption are applied to Remote Service. Furthermore, there is a conflict between Caching and Encryption as data needs to be encrypted before it is cached.

The technical interactions of these concerns are shown in the Concern Interaction Graph (CIG) in Fig. 3. The syntax of the CIG is the same as for standard GRL goal models. Stakeholder concerns are modeled as GRL actors (\circ , e.g., Applicant Concern), use case concerns are modeled as tasks ($\langle \rangle$, e.g., UC Submit Application), and NFR concerns are modeled as softgoals (\square , e.g., NFR Security: Encryption). Correlation links (\dashrightarrow) are used to indicate conflicts between concerns, while GRL dependency links (\dashv) indicate dependencies. The direction of a link indicates priorities among concerns, i.e., the target of the link has higher priority than the source of the link and is therefore applied first (e.g., the Applicant Concern has priority over UC Submit Application).

The GRL elements in Fig. 3 are linked with their corresponding elements in Fig. 2 or stakeholder and use case elements in the AoURN model (e.g., NFR Security: Encryption is linked with Encryption). The satisfaction values of linked elements are synchronized. Use case and NFR concerns are typically set to either 0 or 100 in the CIG, because an evaluation of the goal model investigates a particular set of use case

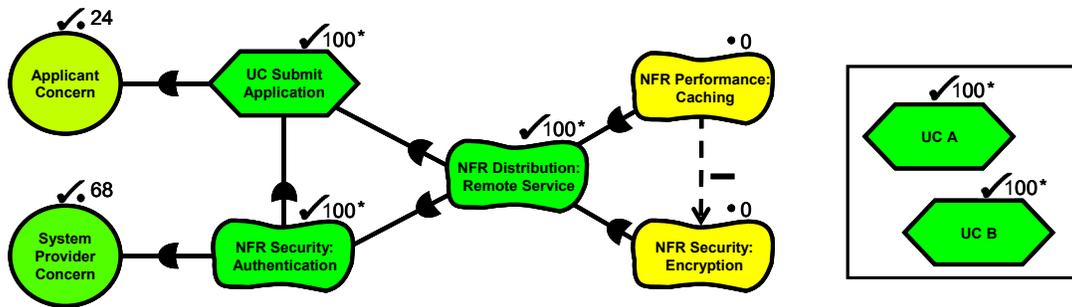


Fig. 3. Concern Interaction Graph

or NFR concerns and therefore sets selected concerns to 100 while others default to 0. The satisfaction values of stakeholder concerns, on the other hand, result from the propagation of those initial values and may therefore have any value. Concerns that are not shown on the CIG do not have any technical interactions with any other concern. Finally, the rectangle in Fig. 3 indicates concerns for which an interaction has been identified but a resolution has not been found.

The evaluation mechanism for the CIG reflects the meaning of the dependency and correlation links in the context of technical concern interactions. For example, if NFR Distribution: Remote Service, UC Submit Application, and NFR Security: Authentication are set to 100, 0, and 0 in Fig. 3, respectively, the evaluation mechanism will change the satisfaction value of NFR Distribution: Remote Service to 0 because its dependencies are all 0 (i.e., the concern cannot be applied because the concerns it depends on are not applied). The changed satisfaction value is fed back to the other goal models and influences the outcome of the evaluation. For example, Remote Service now does not negatively impact Confidentiality and Performance because its satisfaction value is now 0 instead of the original 100 and therefore is not propagated. Alternatively, instead of changing the satisfaction value, an evaluation conflict may be indicated in the CIG.

As the links in the CIG define precedence rules, a composition order for the concerns is established. Composition unfolds in several waves starting with all

concerns that are not restricted by precedence rules (i.e., all CIG nodes that are not the source of any link). Concerns with a satisfaction value of 0 are not considered because this value indicates that they have not been selected for the system. At the end of each wave, the concerns that were applied are considered removed from the CIG, leading to new concerns without outgoing links. These concerns will then be applied in the following wave and so on. Fig. 4 shows the waves for the example in Fig. 3. The use of colors in Fig. 4 is just for illustration purposes and does not carry any meaning. Note that any concerns that are not shown on the CIG are applied with the first wave since these concerns are not restricted by precedence rules by default.

In a different AoURN evaluation where UC Submit Application and NFR Security: Authentication are not selected but NFR Distribution: Remote Service, NFR Performance: Caching and NFR Security: Encryption are selected, the first wave includes the two stakeholder concerns, the second wave NFR Security: Encryption, and the last wave NFR Performance: Caching. NFR Distribution: Remote Service is skipped because its satisfaction value changed from 100 to 0.

While the concrete syntax of the CIG reuses existing GRL symbols (except for the rectangle which is new), the abstract syntax of AoURN is extended to support precedence rules among concerns more explicitly. Two self-associations are added to the Concern metaclass – one for dependencies and one for conflicts.

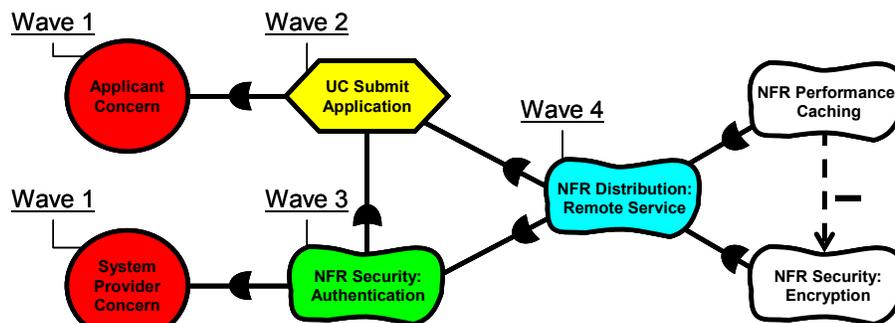


Fig. 4. Waves of composition

5. Conclusion and Future Work

We have presented the Concern Interaction Graph (CIG) that captures the technical interactions among concerns as well their resolutions. The CIG therefore defines the order in which concerns should be composed. Furthermore, the CIG is a goal model that is integrated with other goal models that describe intrinsic interactions as well as the intentions of stakeholders, allowing for a holistic analysis of trade-offs among concerns and stakeholder goals that takes technical and intrinsic interaction into account, while keeping the two types of interactions clearly separated.

We have used CIG on several initial case studies but more experiments will need to investigate circular dependencies among concerns. This could lead to hierarchical CIGs and a more recursive approach for concern composition. Furthermore, the relationship between the CIG and other goal models may be refined. The discovery of aspects from goal models based on the procedure in [23] could be explored, and the later work could benefit from AoURN and CIG to represent aspects and their interactions. Also, mappings other than the one-to-one mapping between CIG elements and elements in the goal model for intrinsic interactions are also possible and should be supported.

Acknowledgements. This research was partially supported by NSERC Canada, through its programs of Discovery Grants and Postgraduate Scholarships.

References

- [1] Mussbacher, G., Amyot, D., Whittle, J., and Weiss, M.: "Flexible and Expressive Composition Rules with Aspect-oriented Use Case Maps (AoUCM)". Moreira, A. and Grundy, J. (Editors), *Early Aspects: Current Challenges and Future Directions*, Springer, LNCS 4765, pp 19-38, 2007.
- [2] Bouma, L.G., Griffith, N., and Kimbler, N. (Editors): "Feature Interactions in Telecommunications Systems". *Computer Networks*, vol. 32(4), 2000.
- [3] ITU-T: *ITU-T Recommendation Z.151: User Requirements Notation (URN) - Language definition*. ITU-T, Geneva, Switzerland, 2008.
- [4] Mussbacher, G., Whittle, J., and Amyot, D.: "Towards Semantic-Based Aspect Interaction Detection". *1st Intl. Wksh. on Non-functional System Properties in Domain Specific Modeling Languages (NFPinDSML2008)*, France, 2008.
- [5] Bakre, S. and Elrad, T.: "Scenario based resolution of aspect interactions with aspect interaction charts". *10th International Workshop on Aspect Oriented Modeling (at AOSD)*, Vancouver, Canada, pp. 1-6, 2007.
- [6] 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, 2007.
- [7] Kienzle, J., Klein, J., and Guelfi, N.: "RAM: Reusable Aspect Models - How to Specify Aspects in the Presence of Dependencies, Variabilities and Conflicts". Submitted to *TAOSD*. http://www.cs.mcgill.ca/~joerg/SEL/RAM_files/taosd2008_RAM_Kienzle.pdf (accessed January 2009)
- [8] Sanen, F., et al.: "Classifying and Documenting Aspect Interactions". *Work. on Aspects, Components and Patterns for Infrastructure Software at AOSD*, Bonn, Germany, 2006.
- [9] Zhang, J., Cottenier, T., Van den Berg, A., and Gray, J.: "Aspect Composition in the Motorola Aspect-Oriented Modeling Weaver". *J. of Object Tech.*, vol. 6, pp. 89-108, 2007.
- [10] Mussbacher, G.: "Aspect-Oriented User Requirements Notation". Giese, H. (Editor), *Models in Software Engineering: Workshops and Symposia at MODELS 2007*, Springer, LNCS 5002, pp 305-316, August 2008.
- [11] Mussbacher, G., Amyot, D., Araújo, J., Moreira, A., and Weiss, M.: "Visualizing Aspect-Oriented Goal Models with AoGRL". *2nd Intl. Workshop on Requirements Engineering Visualization (REV'07)*, New Delhi, India, October 2007.
- [12] Mussbacher, G., Amyot, D., and Weiss, M.: "Visualizing Aspect-Oriented Requirements Scenarios with Use Case Maps". *International Workshop on Requirements Engineering Visualization (REV 2006)*, Minneapolis, USA, Sep. 2006.
- [13] Mussbacher, G., Amyot, D., and Weiss, M.: "Visualizing Early Aspects with Use Case Maps". Rashid, A. and Aksit, M. (Editors), *Trans. on Aspect-Oriented Software Development III*, Springer, LNCS 4620, pp 105-143, 2007.
- [14] Shaker, P. and Peters, D.: "Design-Level Detection of Interactions in Aspect-Oriented Systems". *Aspects, Dependencies and Interactions Workshop at ECOOP 2006*, 2006.
- [15] Douence, R., Fradet, P., and Südholt, M.: "Composition, reuse and interaction analysis of stateful aspects". *Aspect Oriented Software Development*, pp. 141-150, 2004.
- [16] Chitchyan, R., Rashid, A., Rayson, P., and Waters, R.: "Semantics-Based Composition for Aspect-Oriented Requirements Engineering". *Aspect-Oriented Software Development (AOSD)*, Vancouver, Canada, pp. 36-48, 2007.
- [17] Klein, J., Hérouët, L., and Jézéquel, J.-M.: "Semantic-Based Weaving of Scenarios". *Aspect-Oriented Software Development (AOSD)*, Vancouver, Canada, pp. 27-38, 2006.
- [18] Calder, M., Kolberg, M., Magill, E. H., and Reiff-Marganiec, S.: "Feature interaction: a critical review and considered forecast". *Computer Networks*, vol. 41, pp. 115-141, 2003.
- [19] du Bousquet, L. and Richier, J.-L. (Editors): *Feature Interactions in Software and Communication Systems IX (ICFI)*. IOS Press, 2007.
- [20] Metzger, A., Bühne, S., Lauenroth, K., and Pohl, K.: "Considering Feature Interactions in Product Lines: Towards the Automatic Derivation of Dependencies between Product Variants". *ICFI 2005*, Leicester, UK, 2005.
- [21] Weiss, M., Esfandiari, B., and Luo, Y.: "Towards a classification of web service feature interactions". *Computer Networks*, vol. 51(2), 359-381, 2007.
- [22] *jUCMNav*. University of Ottawa; <http://jucmnav.softwareengineering.ca/jucmnav> (accessed January 2009).
- [23] Yu, Y., Leite, J.C.S.d.P., and Mylopoulos, J.: "From Goals to Aspects: Discovering Aspects from Requirements Goal Models". *12th Int. RE Conf. (RE04)*, Kyoto, Japan, 2004.