Aspect-Oriented User Requirements Notation

Gunter Mussbacher

SITE, University of Ottawa, 800 King Edward, Ottawa, ON, K1N 6N5, Canada
gunterm@site.uottawa.ca

1 Research Hypothesis

The Aspect-oriented URN (AoURN) [11] effort extends the User Requirements Notation (URN) [2, 14] with aspects. To date and to the best of our knowledge, no standardized framework unifying goal-oriented, scenario-based, and aspect-oriented concepts exists. The goal of this research is to do this with minimal changes to URN in order to ensure that requirements engineers can continue working with familiar notations while at the same time providing better encapsulation for all types of concerns at the goal and scenario-level (regardless of whether concerns crosscut or not). Aspects can improve the modularity, reusability, scalability, and maintainability of URN models. Considering the strong overlap between non-functional requirements (NFRs) and crosscutting concerns, aspects can help bridge the gap between goals and scenarios. Furthermore, Early Aspects (EA) research can benefit from a standardized way of modeling NFRs (and therefore crosscutting concerns) with AoURN.

2 Introduction to Technical Problem

By the end of the 1990s, Aspect-Oriented Programming (AOP) [8] allowed software engineers to better encapsulate, at the implementation level, crosscutting concerns (i.e. aspects) which are notoriously difficult to modularize with a single dominant modularization technique alone (e.g. with object-oriented concepts). During the last decade, the research community has shifted its emphasis more to Early Aspects (EA) [6] by investigating ways of addressing crosscutting concerns in requirements and design models. Two of the most common requirements engineering models are goal-oriented and scenario-based models. The User Requirements Notation (URN) [2, 14] is the first and currently only standardization effort that combines goal and scenario models in one language. The Aspect-oriented URN (AoURN) [11] aims to extend URN with aspect concepts.

3 Overview of Related Work

3.1 User Requirements Notation

The User Requirements Notation (URN) [2, 14], a standardization effort of the International Telecommunication Union (ITU-T Z.150 Series), contains two complementary modeling languages for goals and scenarios. The Goal-oriented Requirement Language (GRL) is a visual modeling notation for business goals and non-functional requirements (NFRs) of many stakeholders, for alternatives to be considered, for decisions that were made, and for rationales that helped make these decisions. GRL supports reasoning about goals and NFRs with the help of GRL strategies and an evaluation mechanism that propagates low-level decisions regarding alternatives to satisfaction ratings of high-level stakeholder goals and NFRs. Use Case Maps (UCMs) are a visual scenario notation that focuses on the causal flow of behavior superimposed on a structure of components. UCMs depict the interaction of architectural entities while abstracting from message and data details. UCMs support the definition of scenarios including pre-conditions and post-conditions. A scenario describes a specific path through the UCM model where only one alternative at any choice point is taken. Furthermore, URN allows URN links to be established between any modeling elements. Over the last decade, GRL and UCMs have successfully been used not just for telecommunication systems but in many application domains [14].

3.2 Early Aspects

As the Aspect-oriented User Requirements Notation (AoURN) [11] makes use of goal and scenario models, we will briefly review aspect-oriented approaches to requirements engineering that apply to goal and scenario models. For a comparison of these approaches to AoURN or an introduction to aspects, see [10–12].

In *Aspect-Oriented Software Development (AOSD) with Use Cases* [4], Jacobson and Ng consider a well-written use case a concern. Extension points identify a step in a use case where an extension may occur. Pointcuts in other use cases reference such extension points. Aspects allow use cases to be encapsulated throughout the software development lifecycle.

In *Scenario Modeling with Aspects* [4], Whittle and Araújo model aspectual scenarios with sequence-diagram-like *interaction pattern specifications* (IPS) and *state machine pattern specifications* (SMPS). IPS and SMPS define roles which can be bound to elements in other sequence diagrams and state machines.

In the *Aspectual Use Case Driven Approach* [4], Araújo and Moreira visualize how crosscutting non-functional requirements captured with templates are linked to functional requirements (use case diagrams or sequence diagrams). *Activity pattern specifications* (APS) similar to the aforementioned IPS and SMPS are used. In addition, new use-case relationships allow the impact of one use case on another to be described (restricting or contributing positively/negatively).

Barros and Gomes [3] apply aspect-orientation to activity diagrams (AD) by describing ways to merge stereotyped nodes in one AD with nodes in another.

Whittle *et al.* [15] propose a metamodel-based aspect composition technique that uses graph transformation formalisms. This approach can be applied to any model for which a metamodel has been defined.

In the Use Case Map (UCM) community, de Bruin and van Vliet [5] allow behaviour to be added before and after a UCM by explicitly adding “Pre” and “Post” stubs (placeholders for sub-maps) to the UCM.

Yu *et al.* [16] identify aspects in goal models based on relationships between functional and non-functional goals. Goal aspects are proposed to address scalability issues but it is pointed out that the goal aspects' syntax still requires further research.

Alencar *et al.* [1] identify aspects in i* models. Their extensions to aspect-oriented concepts, however, do not fully separate concerns from other concerns.

Kaiya and Saeki [7] propose a pattern-based technique to compose view-points. The approach lacks formalization and limits its composition to a simple combinatorial approach instead of more powerful pointcut expressions.

4 Sketch of Proposed Solution

AoURN extends URN by defining a *joinpoint model* for the Goal-oriented Requirements Language (GRL) and Use Case Maps (UCMs). All nodes of GRL graphs and UCMs (with a few exceptions) are joinpoints that can be matched by pointcut expressions. Pointcut expressions are defined on *pointcut graphs and maps* that are matched against the rest of the model. Pointcut graphs and maps are standard URN diagrams that can also be parameterized for increased matching power by allowing names of modeling elements to contain wildcards and logical expressions. The goals, behavior, and structure of aspects are defined on *advice graphs and maps* which are loosely coupled to pointcut graphs and maps. Flexible composition rules defined on advice graphs and maps are described with URN itself and are therefore as expressive as URN and not restricted by the capabilities of any particular pointcut language (which for example could only allow standard before/after/around rules).

AoURN's ability to encapsulate NFRs (major concerns in goal models) in both model types and also use cases (major concerns in scenario models) in both model types bridges the gap between goal and scenario models. This gap is further narrowed by URN traceability links between modeling elements of goal and scenario aspects.

5 Anticipated Contributions

Work items 1-6 related to the fundamental infrastructure for AoURN are nearing completion. While early results exist for work items 7-10, further research is required and expected to be completed next year. The remaining items describe further avenues of research.

1. Extension of the abstract syntax, the concrete syntax, and the semantics of URN with aspect-oriented concepts
2. Unification of goal/scenario/aspect concepts in one framework
3. Clarification of the relationship between aspects in goal-oriented models and aspects in scenario-based models within AoURN
4. Composition/visualization of aspects based on the notion of dynamic stubs (a UCM element that can possibly be generalized to other languages)
5. Flexible composition of aspects only limited by the expressiveness of URN itself (as opposed to a particular pointcut or composition language)

6. Clarification of the semantics of URN and thus AoURN (as pointcut matching and composition of aspects require precise semantics)
7. Proof of concept by providing tool support for AoURN with jUCMNav [13]
8. Qualitative assessment of AoURN with respect to desirable properties of aspect-oriented requirements models
9. Quantitative assessment of AoURN based on metrics for aspect-oriented requirements models adapted for URN and AoURN
10. Clarification of the relationship of aspects, GRL strategies, and UCM scenarios
11. Cataloguing of the common kinds of crosscutting for scenarios and goals as crosscutting patterns (e.g. interleaving of scenarios)
12. Investigating the applicability of advanced URN research to AoURN (e.g. feature interaction, business process modelling, performance analysis, product lines, or modeling inherently existing communication aspects in UCM)

6 Progress, Methods, and Evaluation

The groundwork for the extension of URN with aspect-oriented concepts has been laid. A journal paper [11] and workshop papers [10, 12] on Aspect-oriented Use Case Maps (AoUCM) as well as UCM semantics [9] have been accepted and the first portions of an AoUCM prototype are now being included into the official jUCMNav [13] release with the rest following shortly. Papers on the Aspect- and Goal-oriented Requirements Language (AoGRL) have been submitted to conferences. These papers cover the fundamental infrastructure work mentioned in Sect. 5 and introduce AoURN including its algorithms for pointcut matching and aspect composition. Furthermore, the papers present small case studies as well as initial qualitative and quantitative assessments of AoURN.

In general, the research is being evaluated with three methods. First, support for AoURN is implemented in the open-source tool jUCMNav (the most popular URN editor) to ensure the feasibility of the proposed notation. Second, AoURN and URN models for the same system are created with the tool as part of a case study. Third, qualitative and quantitative assessments of the models are performed in order to compare AoURN with URN and measure any improvements. The metrics used for the quantitative assessments have been adapted from literature for URN and AoURN models by mapping the notion of component to aspect, goal graph, and map as well as the notion of operation to select URN nodes (intentional element, responsibility, stub). The metrics measure separation of concerns, coupling, lack of cohesion, and size. The metrics so far show that the complexity of goal graphs and maps in URN is traded against the complexity of pointcut graphs/maps in AoURN. Overall, however, AoURN outperforms URN based on the results. Furthermore, AoURN is compared qualitatively with other aspect-oriented techniques. At this stage, AoUCM has been compared with other aspect-oriented approaches to scenario-based requirements engineering with respect to the following properties: obliviousness, exhaustive composition rules, scalability, familiarity, formality, appropriate abstraction level, and reusability.

The results rank AoUCM and metamodel-based aspect composition techniques using graph transformation formalisms at the top. While AoUCM is only a semi-formal notation, graph transformation techniques suffer from lack of familiarity. Finally, feedback on the proposed research is received through collaboration with established researchers in the Early Aspects (EA) field and participation in international conferences, workshops, tutorials, etc.

References

1. Alencar, F., Moreira, A., Araújo, J, Castro, J., Silva, C., and Mylopoulos, J.: Towards an Approach to Integrate i* with Aspects. *8th Intl. Bi-Conf. Wksh. on Agent-Oriented Inf. Systems (AOIS-2006)* at CAiSE'06, Luxembourg (June 2006)
2. Amyot, D.: Introduction to the User Requirements Notation: Learning by Example. *Computer Networks*, Vol. 42(3), pp 285-301 (21 June 2003)
3. Barros, J.-P. and Gomes, L.: Toward the Support for Crosscutting Concerns in Activity Diagrams: a Graphical Approach. *Workshop on Aspect-Oriented Modelling* (held with UML 2003), San Francisco, California, USA (October 2003)
4. Chitchyan, R. et al.: *Survey of Analysis and Design Approaches.* AOSD-Europe Report ULANC-9 (May 2005), http://www.aosd-europe.net/deliverables/d11.pdf (acc. June 2007, see 3.4.3 for more detailed references to scenario-based approaches)
5. de Bruin, H. and van Vliet, H.: Quality-Driven Software Architecture Composition. *Journal of Systems and Software*, Vol. 66(3), pp 269-284 (15 June 2003)
6. *Early Aspects website.* http://www.early-aspects.net/ (accessed May 2007)
7. Kaiya, H. and Saeki, M.: Weaving Multiple Viewpoint Specifications in Goal-Oriented Requirements Analysis. *11th Asia-Pacific Software Eng. Conf. (APSEC 2004)*, Busan, Korea, IEEE Computer Society Press, pp 418-427 (Nov. 2004)
8. Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J.-M., and Irwin, J.: Aspect-Oriented Programming. *ECOOP'97–Object Oriented Programming, 11th European Conference*, LNCS 1241, pp 220-242, Springer (1997)
9. Mussbacher, G.: Evolving Use Case Maps as a Scenario and Workflow Description Language. *10th Wksh. on Rqmnts. Eng. (WER 2007)*, Toronto, Canada (May 2007)
10. 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 (September 2006)
11. Mussbacher, G., Amyot, D., and Weiss M.: Visualizing Early Aspects with Use Case Maps. *To appear in Transactions on Aspect-Oriented Software Development.*
12. Mussbacher, G., Amyot, D., Whittle, J., and Weiss M.: Flexible and Expressive Composition Rules with Aspect-oriented Use Case Maps (AoUCM). *10th International Wksh. on Early Aspects (EA 2007)*, Vancouver, Canada (March 13, 2007)
13. Roy, J-F.: *Requirements Engineering with URN: Integrating Goals and Scenarios.* MSc. thesis, OCICS, University of Ottawa, Canada (2007), http://www.softwareengineering.ca/jucmnav (accessed May 2007)
14. *URN Virtual Library.* http://www.usecasemaps.org/pub (accessed June 2007)
15. Whittle, J., Araújo, J., Moreira, A., and Rabbi, R.: Graphical Composition of State-Dependent Use Case Behavioral Models. *ISE Dptmnt. Techn. Rep.*, George Mason Univ., ISE-TR-07-01. http://ise.gmu.edu/techrep/2007 (acc. May 2007)
16. Yu, Y., Leite, J. C. S. d. P., and Mylopoulos, J.: From Goals to Aspects: Discovering Aspects from Requirements Goal Models. *12th International Requirements Engineering Conference (RE'04)*, Kyoto, Japan (September 2004)