The Aspect-Oriented User Requirements Notation: Aspects, Goals, and Scenarios

Gunter Mussbacher
Department of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive
Ottawa, K1S 5B6, Canada
gunter@sce.carleton.ca

ABSTRACT
This tutorial discusses aspect-oriented requirements engineering, focusing on scenario-based and goal-oriented requirements models with the Aspect-oriented User Requirements Notation (AoURN). AoURN is an extension of the User Requirements Notation (URN), a recent international modeling standard for requirements engineering published by the International Telecommunication Union. AoURN is a strong candidate for inclusion in future versions of the standard. While the tutorial gives a thorough introduction to AoURN, it places particular emphasis on AoURN's advanced composition mechanisms which enable interleaved and semantics-enhanced compositions. In addition, the impact of aspect-orientation on existing URN analysis capabilities is discussed.

Categories and Subject Descriptors
D.2.1 [**Software Engineering**]: Requirements/Specifications – *languages, methodologies, tools.*

General Terms
Standardization, Languages.

Keywords
Aspect-oriented User Requirements Notation, AoURN, URN, Use Case Maps, UCM, Goal-oriented Requirement Language, GRL.

1. TUTORIAL OVERVIEW
The User Requirements Notation (URN) [1] is a recent modeling standard, published by the International Telecommunication Union in 2008, that supports the elicitation, analysis, specification, and validation of requirements. URN is the first standard that combines modeling concepts and notations for goals and intentions (mainly for non-functional requirements (NFRs), quality attributes, and reasoning about alternatives) and scenarios (mainly for operational requirements, functional requirements, and performance and architectural reasoning). Changes to requirements are inevitable and ideal requirements models should cope with change efficiently. However, crosscutting requirements make changing requirements more difficult, since change impact is more complicated and costly to manage due to scattered and tangled requirements models. Therefore, it is necessary even at the requirements stage to model crosscutting concerns (aspects) independently, but also to compose aspects with other requirements in a way that will allow the entire set of requirements to be analyzed and validated.

This tutorial discusses aspect-oriented requirements models. The tutorial concentrates on goal-based and scenario-based requirements engineering with the Aspect-oriented URN (AoURN) [3][4][5]. AoURN is an extension to the URN standard and a strong candidate for inclusion in future versions of the standard as it aims to address the aforementioned issues. AoURN allows for a thorough introduction to aspect-oriented requirements engineering, as goals and scenarios are commonly used requirements engineering techniques and AoURN contains two very different modeling notations for goals (AoGRL – Aspect-oriented and Goal-oriented Requirement Language) and for scenarios (AoUCM – Aspect-oriented Use Case Maps) that require tailored approaches for aspect-oriented modeling. AoGRL has concepts for the specification of stakeholders, their goals, NFRs of interest, rationales, and potential solutions, whereas AoUCM has concepts for the specification of behavior, scenarios, and structuring that describe potential solutions in more detail. Major concerns modeled with AoURN are typically stakeholder goals, NFRs, and use cases/scenarios.

As AoURN expresses concern composition rules with URN itself, it is possible to describe rules in an exhaustive and highly flexible way that is not restricted by any specific composition language. AoURN's support for composition goes well beyond the usual before/after/replacement options and, e.g., includes parallel, alternative, loop, and interleaved composition. The tutorial places particular emphasis on AoURN's advanced composition mechanisms that allow for interleaved and semantics-enhanced compositions. Interleaved composition merges two scenarios together while respecting the ordering of steps in both scenarios. Semantics-enhanced composition takes into account (i) model elements that represent "whitespace" and (ii) hierarchical structuring. "Whitespace" elements exist only as visual aids for the modeler but are otherwise semantically insignificant. In terms of hierarchical structuring, AoURN's composition mechanism ensures that an AoURN model is matched regardless of whether the model is described in one hierarchical layer or with the help of many hierarchical layers. Consequently, refactoring operations that change the hierarchical structuring of an AoURN model may be performed with the guarantee that all concerns are applied the same way before and after refactoring. Furthermore, semantic-enhanced composition is also applicable when different syntax may be used to describe

AOSD'11, March 21–25, 2011, Pernambuco, Brazil.
ACM 978-1-4503-0606-5/11/03.

semantically equivalent models. In these cases, both syntax variations are matched by AoURN's composition mechanism.

Tool support for AoURN is provided through jUCMNav [2], an open-source Eclipse plug-in developed at the University of Ottawa. jUCMNav supports full concern management (see Figure 1), the specification of AoUCM models (see right side of Figure 2), to a limited extent the specification of AoGRL models, the composition of most AoUCM models (see left side of Figure 2), as well as the navigation of aspect-oriented models. As of now, composition of AoGRL is not yet supported.

The definition of aspect-oriented models, however, is only a first step that needs to be complemented by aspect-oriented approaches for any existing analysis and validation techniques. Consequently, the built-in URN analysis techniques for stakeholder trade-offs and scenario test suites are discussed from an aspect-oriented point of view, thus presenting a road-map on how to address these techniques in the AoURN context. For more information about URN and AoURN, the reader is referred to the URN Virtual Library [6].

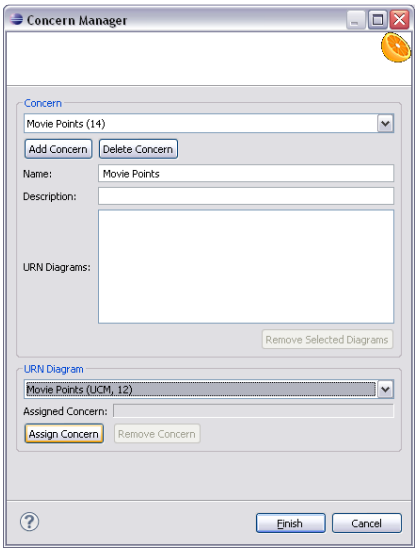

Figure 1. jUCMNav: Concern Management

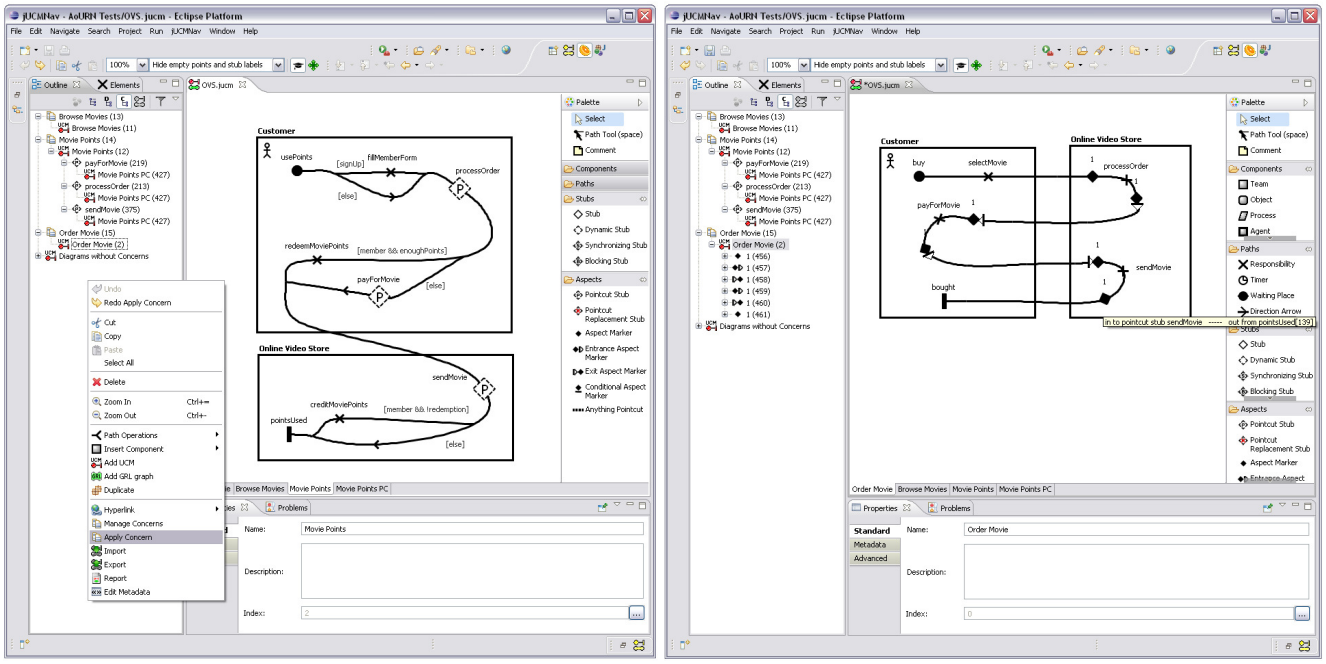

Figure 2. jUCMNav: AoUCM Specification and Composition

2. ACKNOWLEDGMENTS

This tutorial was supported by the NSERC Postdoctoral Fellowships program.

3. REFERENCES

[1] International Telecommunication Union (ITU-T) 2008. *Recommendation Z.151 (11/08): User Requirements Notation (URN) - Language Definition*. Geneva.

[2] *jUCMNav website*. Version 4.3.0, University of Ottawa; http://softwareengineering.ca/jucmnav (accessed Feb. 2011)

[3] Mussbacher, G., Amyot, D., Araújo, J., and Moreira, A. 2010. Requirements Modeling with the Aspect-oriented User Requirements Notation (AoURN): A Case Study. *Transactions on Aspect-Oriented Software Development (TAOSD) VII*, LNCS 6210 (2010), 23–68. DOI= http://dx.doi.org/10.1007/978-3-642-16086-8_2.

[4] Mussbacher, G. and Amyot, D. 2009. Extending the User Requirements Notation with Aspect-oriented Concepts. *14th SDL Forum (SDL 2009)*, Bochum, Germany, September 2009. Reed, R., Bilgic, A., Gotzhein, R. (Eds.), *SDL 2009: Design for Motes and Mobiles*, Springer, LNCS 5719 (2009), 115–132. DOI= http://dx.doi.org/10.1007/978-3-642-04554-7_8.

[5] Mussbacher, G. 2010. *Aspect-oriented User Requirements Notation*. PhD thesis, University of Ottawa, Canada.

[6] *URN Virtual Library*. http://www.usecasemaps.org/pub (accessed February 2011)