11th International Workshop on Aspect-Oriented Modeling

Jörg Kienzle[1], Jeff Gray[2], Dominik Stein[3], Walter Cazzola[4], Omar Aldawud[5], Tzilla Elrad[6]

[1]McGill University, Canada; [2]University of Alabama at Birmingham, USA; [3]University of Duisburg-Essen, Germany; [4]University of Milano, Italy; [5]Lucent Technologies, USA; [6]Illinois Institute of Technology, USA
joerg.kienzle@mcgill.ca, gray@cis.uab.edu, dominik.stein@icb.uni-due.de, cazzola@dico.unimi.it, oaldawud@lucent.com, elrad@iit.edu

Abstract. This report summarizes the results and discussions from the *11th Workshop on Aspect-Oriented Modeling (AOM)*. The workshop was held in conjunction with the *International Conference on Model-Driven Engineering, Languages, and Systems (MODELS)*, which was located in Nashville, Tennessee, on September 30, 2007. Over 20 researchers and practitioners attended the workshop with various backgrounds in aspect-oriented software development and software model engineering. The workshop provided a forum for discussing the state of the art in modeling crosscutting concerns at different stages of the software development process: requirements elicitation and analysis, software architecture, detailed design, and mapping to aspect-oriented programming constructs. This workshop summary provides an overview of the accepted submissions and summarizes the results of the different discussion groups. Papers, presentation slides, and photos from the workshop are available at http://www.aspect-modeling.org/.

1 Introduction

This brief summary reports on the outcomes of the *11th International Aspect-Oriented Modeling Workshop*. The workshop took place at the Marriott Hotel in Nashville, Tennessee, on Sunday, September 30, 2007. The workshop was part of the *10th International Conference on Model Driven Engineering Languages and Systems – (MODELS 2007)*. A total of 10 position papers were submitted and reviewed by the program committee, 7 of which were accepted to the workshop. Over 20 participants attended the presentation session and took part in afternoon working group discussions. Papers, presentation slides, and further information can be found at the workshop website, which is at http://www.aspect-modeling.org/. The website also has links to the previous editions of the workshop.

The rest of this report is structured as follows: Section 2 provides a general overview of the motivation, goals, and challenges of aspect-oriented modeling. Section 3 gives a summary of the papers that have been accepted to this workshop. Section 4 outlines the results of the discussion groups. Finally, section 5 concludes the report.

2 Overview to Aspect-Oriented Modeling

Aspect-orientation is a rapidly advancing technology. New and powerful aspect-oriented programming techniques are presented at many international venues every year. However, aspect-oriented software development techniques are often deeply-rooted in the "intimate essence" of a program, i.e., the syntax and structure of its code. As a consequence, developers may easily be overwhelmed with implementation details, while loosing track of the intentions and goals of the interacting (formerly crosscutting) concerns, as well as of where and how they interact. Aspect-oriented modeling has the potential to provide the necessary tool for abstracting from the essence of the problem and for taking root in the semantic nature of the interacting concerns and their interaction.

Over the last five years, much research work has been presented at the various editions of this workshop, which all aim at helping developers not to get lost in the "code space" and its associated accidental complexities. Consolidating that work, three important fields of research have emerged that are frequently and recurrently tackled by different researchers.

One major field of research is concerned with finding appropriate modeling abstractions for aspect-oriented programming language constructs, such as pointcuts, advice, introductions, stateful aspects, aspect-oriented hooks and connectors. The options are usually to define a new modeling language or modeling notation, or to extend an existing one. The goal is to provide aspect-oriented software developers with appropriate means to facilitate the analysis of their problems as well as the design of their solutions. The challenge is to find the right level of abstraction so that the invented modeling means are suitable for a variety of problems as well as for a variety of aspect-oriented implementation techniques (e.g., languages and frameworks).

Another major field of research is concerned with bringing forth the benefits of aspect-orientation to the modeling level and, thus, to model-driven development. Much work in this field of research is concerned with model transformation, model composition, and/or model weaving. One key question is to determine the similarities, differences and relationships between these terms and techniques. A frequent issue that emerges is to what extent conventional transformation techniques can be used to implement aspect-oriented model weavers. The ultimate goal is to free aspect-oriented software developers from the need of using aspect-oriented programming languages.

Yet another field for research that attained much research efforts recently is the specification of validation and verification frameworks for aspect-oriented software. Key concerns in this field include the detection of conflicts between different aspects, the confirmation whether or not aspects affect the correct points in the base application, and the disclosure of the presence of crosscutting in software. Some of the key questions are focused on the formalisms that should be used, how such formalisms can be adopted to aspect-orientation, and how its application can be facilitated for the ordinary aspect-oriented software developer.

These three core research concerns represent a non-exclusive list. Multiple researchers have addressed many other issues. Several of the papers presented at the current workshop edition fall into the second category. The workshop discussions considered all of these issues and other concerns, as noted in the next two sections.

3 Summary of Accepted Position Papers

Jon Whittle from Lancaster University, UK, presented MATA [1], an approach to aspect-oriented modeling that can be applied to any modeling language with a well-defined meta-model. MATA provides expressive composition mechanisms. The weaving in MATA is based on graph transformations, and hence any composition technique is possible. MATA also provides support for detecting some aspect interactions automatically. The MATA tool is implemented as a bridge between IBM Rational Software Modeler (RSM) and Attributed Graph Grammar System (AGG).

Frank Fleurey from IRISA/INRIA in France presented a generic approach for automatic model composition [2]. In the approach, the two models that are to be composed are first pre-processed, then merged, and finally the resulting model is post-processed. The composition is signature-based: if a signature matches, the elements are merged and recursively processed; in case there is no match, both elements are copied to the target model. He presented a reusable composition framework, in which a meta-model is extended to obtain a "composable meta-model": mergeable model elements have to define *getSignature* operations that allow the composition algorithm to match them. Special signatures have to be defined that make sense for each model element.

Hua Xiao from Queens University, Canada, presented an approach to weave business requirements into model transformations [3]. He argued that there is a gap between the business domain, the concerns of business stakeholders, and the technical domain, as considered by the developers. With their work, the authors want to help to integrate business requirements into generic design models and implementation models. BPEL (Business Process Engineering Language) is too limited to represent all requirements specified by the business analysts. The authors use AOM techniques to enhance a primary BPEL model with other concerns such as time, cost, resource usage, performance, and frequency. A weaver combines the BPEL and aspect models to yield a composed model, which is wrapped and can then be fed to a simulation engine. After the simulation, the developer can validate the successful achievement of business requirements (e.g., compare revenue and cost per request).

Stefan Van Baelen from the Katholieke Universiteit Leuven, Belgium, presented an approach that composes application models and security models [4]. He highlighted the fact that security concerns are very spread out through an application and cannot easily be isolated in an application layer. He compared the advantages and disadvantages of the classical AOP approach, the MDD approach (a security-independent application model is transformed to a security-aware application model, then mapped to an execution platform), and the AOM approach: generate a woven model, and then generate OOP code. They also support generation of CeasarJ code. Their approach allows the definition of company-wide access policies, which can be woven into many applications. At run-time, the application contacts an authorization engine to check access rights. The sample application he showed defined concepts with a UML class diagram. Access policy subjects and objects are also represented in UML class diagrams, and then "merged" or assigned to each other.

Jaime Pavlich-Mariscal from the University of Connecticut, USA, presented his position paper on how to enhance UML to model custom security aspects [5]. He first presented different access control models (RBAC, MAC and DAC), and highlighted

that UML does not explicitly support any of them. Also, it is not easily possible to use capabilities of different access control models in combination. As a result, traceability of access control policies is also difficult to achieve. Their approach adds four security-specific models, which are then composed with the application model to create a security-aware model. Security features that can be modeled include: positive/negative permissions, MAC or delegation rules. Model composition is achieved by merging meta-models.

Jörg Kienzle from McGill University, Canada presented an aspect-oriented modeling approach for specifying reusable aspect models [6]. His aspect models define structure using class diagrams, and behavior using sequence diagrams. The weaver, based on existing class diagram and sequence diagram weavers, is capable of weaving aspects with other aspects or base models. In his talk, Jörg demonstrated the high degree of reusability of the aspect models by modeling the design of 8 inter-dependent aspects of the *AspectOptima* case study, an aspect framework that implements support for transactions. Based on this experience, he identified several modeling language features that seem essential to support reusable aspect modeling.

Steffen Zschaler from the University of Dresden, Germany, presented a talk entitled "Aspect-Orientation for your Language of Choice" [7]. He presented *Reuseware*, a tool based on invasive software composition that makes it possible to define fragments of models, where each fragment can define interfaces in forms of slots. Later, elements identified as anchors within one model can be bound to the slots in another model using queries based on pattern-matching expressions.

4 Overview of Discussion Topics

This section offers a summary of the most interesting and significant issues that were addressed during the discussion sessions. These issues also emerged during the questions and comments in the presentation sessions.

What are the characteristics of a good aspect modeling language? Participants in the aspect composition group considered this question and stated that a good aspect modeling language would have a rigorous semantic definition with a mixture of external and internal behavior descriptions. A question arose in the discussion regarding whether the aspect modeling language needed to be similar to the base modeling language. If the modeled concern is close to the domain, there may be benefits in having the aspect and base modeling languages similar.

Should an aspect modeling language be standardized? There was an overwhelming consensus that it is too early to have a standardized aspect modeling language. At this stage of AOM maturity, standardization may be too restrictive because no aspect language fits all concerns. An aspect modeling language that is too general may lose its usefulness. Dedicated formalisms can have specific meaning, purpose, and analysis capabilities.

What is the current status of model composition languages to address crosscutting concerns? Current model composition languages are very low-level and force a model engineer to bind too early in the life cycle. Precise bindings may be uncertain at early stages of development (e.g., there may not be clearly defined connections between early aspects and aspects in design). There is a need to model composition relationships over several abstraction levels. Some of the questions that remain to be answered include: Is a simple linear progression through the development phases too naïve? How soon should the aspect binding be realized? How to manage relationships in aspect models (e.g., new relationships appearing and old relationships disappearing)?

What modeling language features enable the creation of reusable aspect models? A reusable aspect model should not refer to any base model element directly, or prescribe a binding to a particular base model element, or depend on the existence of a specific base element. A modeling formalism supporting reuse must provide means to define reusable aspect models in a base model independent way. Mappings between the reusable aspect model and the base should be established in separate bindings. In order to create reusable aspect libraries, the modeling formalism must be able to provide a means to describe aspect specifications, as well as the contexts in which the reusable aspect can be applied. Ideally, a means for detecting conflicts among reusable aspects when they are applied to the same base model should be provided as well.

5 Concluding Remarks

The workshop continued the tradition of having a very diverse representation of participants. The authors came from nine different countries (Argentina, Belgium, Canada, France, Germany, Luxembourg, Netherlands, United Kingdom, and the United States.); likewise, the organizing and programming committees represented eleven countries (Belgium, Canada, France, Germany, Ireland, Italy, Netherlands, Portugal, Switzerland, United Kingdom, and the United States.). In addition to the geographical diversity, the AOM workshop also attracted participants with wide research interests in aspects across the entire spectrum of the development lifecycle. As a result, this provided opportunities for a variety of opinions that were well-informed from the accumulated experience of the participants.

The workshop provided a forum for presenting new ideas and discussing the state of research and practice in modeling various kinds of crosscutting concerns at different levels of abstraction. The workshop identified and discussed the impacts of aspect-oriented technologies on model engineering to provide software developers with general modeling means to express aspects and their crosscutting relationships onto other software artifacts. This workshop report is a summary of the activities for those who could not attend the workshop and provides an opportunity to gain insights to these issues, and to initiate a future research agenda in the field of aspect-oriented modeling.

Acknowledgements

We would like to thank the program committee members who have helped to assure the quality of this workshop: Mehmet Aksit (University of Twente), Aswin van den Berg (Motorola Labs), Thomas Cottenier (Illinois Institute of Technology), Sudipto Ghosh (Colorado State University), Stefan Hanenberg (University of Duisburg-Essen), Andrew Jackson (Trinity University), Jean-Marc Jézéquel (IRISA), Kim Mens (University Catholic of Louvain), Alfonso Pierantonio (University of Aquila), Awais Rashid (Lancaster University), Raghu Reddy (Rochester Institute of Technology), Ella Roubtsova (Open University, Netherlands), Bedir Tekinerdogan (University of Twenty), Markus Völter (Independent Consultant), and Jon Whittle (Lancaster University).

We also thank all submitters and workshop participants who helped to make this workshop a success.

List of Accepted Position Papers

[1] Jon Whittle and Praveen Jayaraman, *MATA: A Tool for Aspect-Oriented Modeling based on Graph Transformation*

[2] Franck Fleurey, Benoit Baudry, Robert France and Sudipto Ghosh, *A Generic Approach For Automatic Model Composition*

[3] Ying Zou, Hua Xiao, Brian Chan, *Weaving Business Requirements into Model Transformations*

[4] Aram Hovsepyan, Stefan Van Baelen, Koen Yskout, Yolande Berbes, Wouter Joosen, *Composing Application Models and Security Models: On the Value of Aspect- Oriented Technologies*

[5] Jaime Pavlich-Mariscal, Laurent Michel, and Steven Demurjian, *Enhancing UML to Model Custom Security Aspects*

[6] Jacques Klein and Jörg Kienzle, *Reusable Aspect Models*

[7] Florian Heidenreich, Jendrik Johannes, and Steffen Zschaler, *Aspect Orientation for Your Language of Choice*