Computer Science
Technical Report

Modeling Approach Comparison Criteria for MODELS 2011 CMA Workshop

Geri Georg, Colorado State University, USA
Gunter Mussbacher, Carleton University, Canada
Betty Cheng, Michigan State University, USA
Ana Moreira, Universidade Nova de Lisbo, Portugal
Robert France, Colorado State University, USA
`{georg,france}@cs.colostate.edu,`
`gunterm@site.uottawa.ca,`
`chengb@cse.msu.edu,`
`amm@di.fct.unl.pt`

September 15, 2011

Technical Report CS-11-104

Computer Science Department
Colorado State University
Fort Collins, CO 80523-1873

Phone: (970) 491-5792   Fax: (970) 491-2466
WWW: http://www.cs.colostate.edu

Modeling Approach Comparison Criteria for MODELS 2011 CMA Workshop

Colorado State University Technical Report CS-11-104

Geri Georg[1], Gunter Mussbacher[2], Betty Cheng[3], Ana Moreira[4], Robert France[1]

[1]Computer Science Department, Colorado State University, Fort Collins, Colorado, USA
[2] Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
[3]Michigan State University, East Lansing, Michigan, USA
[4] Universidade Nova de Lisbo, Lisbon, Portugal

September, 2011

The criteria described in this document represent a work-in-progress, begun at the Bellairs AOM workshop in April 2011. Participants in the workshop brought AO (aspect-oriented) and also OO (object-oriented) modeling expertise. One of the workgroups began a study to define an ontology comprising key concepts of both AO and OO modeling techniques in order to provide a basis for understanding, analysis, and comparison of various approaches.

The workgroup recognized early on that many concepts have slightly different interpretations based not only on whether they are being used to describe an AO or OO context, but also based on their use in a specific phase of software development, or based on a particular viewpoint of the researcher. The group therefore began by soliciting concept descriptions from multiple researchers with a diverse set of interests. The goal of this step is to bring to the community an appreciation of the subtle nuances of the concepts based on their context and to provide a framework for discussion of the key concepts.

The descriptions in this document are the first step in collecting this information, and are meant to provide a basis for describing AO and OO modeling techniques at the MODELS 2011 Comparing Modeling Approaches (CMA) workshop.

We ask that each CMA workshop participant use these descriptions to categorize all of the submissions that were accepted to the workshop as a starting point for discussions. Please use the provided *assessment form* and send the completed form to cma2011-submissions@site.uottawa.ca by October 10, 2011. In cases where key concept descriptions are fairly straightforward and general consensus has been reached on variations, we have provided the variations in a multiple choice format to simplify the categorization process. However, please add new variations and notes as needed.

Key Concepts and Dimensions

Key concepts related to a modeling technique are concepts that are targeted for optimization or improvement by the technique. They thus become the criteria by which a technique can be evaluated, and their description/definition is critical to common understanding between modelers. Please note that these concepts and their use as criteria can be applied to either a technique or to the model(s) that result from using a technique.

Key concepts of a modeling approach can be described along several different dimensions. Thus, some or all of the key concepts may be applicable and in fact have different interpretations depending on the dimension along which they are applied. While we recognize this fact, we present the dimensions and key concepts in a more orthogonal view.

1. Modeling Approach Dimensions

Phase/activity:

During which software development activity is the modeling technique most applicable? Clearly multiple activities may be appropriate for this categorization, e.g., verification of high-level design during evolution.

☐ Early requirements	☐ Late requirements	☐ Architectural
☐ High-level design	☐ Low-level design	☐ Implementation
☐ Deployment	☐ Evolution	☐ Validation / Verification

☐ Other: ____

Notation/language:

A model must be defined in a language. In some cases, models may be expressed in a single language, but there are certainly situations that can benefit from the use of languages that are specific to the model viewpoint (e.g., using a language such as POLICY to define security policies).

What languages/notations are used to define models using this technique?

☐ Standard	versus	☐ Non-standard
☐ Domain specific	versus	☐ General Purpose
☐ Formal Semantics	versus	☐ No Formal Semantics

☐ Other: ____

Application domain:

The application domain refers to the type of system that is being modeled. Examples of system types are automotive, aeronautical, healthcare, transportation.

What, if any, domains are best modeled by the approach?

____

Types of concerns:

The types of concerns that an approach can model are higher-order concepts like themes, aspects, use cases, data, goals, qualities, stakeholders, states, features, classes, etc.

What types of concern can be modeled by the approach?

☐ Themes	☐ Aspects	☐ Use Cases	☐ Data
☐ Goals	☐ Qualities	☐ Stakeholders	☐ States
☐ Features	☐ Classes		

☐ Other: ____

Views supported:

A model can be perceived from multiple perspectives. For many models, there is at least a structural and a behavioral perspective, and these perspectives may be described using different languages/notations (e.g., the structural perspective may be described using class diagrams while the behavioral perspective may be described using state machines or activity diagrams).

What views are supported by the modeling technique?

☐ Structural	☐ Behavioral	☐ Intentional

☐ Other: ____

2. Key Concepts

The key concepts listed below can be applied to different modeling approaches such as AO and OO, but they make take on a different interpretation depending on the context in which they are applied.

Modularity:

The Separation of Concerns principle states that a given problem involves different kinds of concerns, which should be identified and separated to cope with complexity, and to achieve the required engineering quality factors such as robustness, adaptability, maintainability, and reusability. The ideal decision regarding which concerns must be separated from each other, requires finding the right compromise for satisfying multiple (quality) factors. A multi-dimensional decomposition, or as it is called multi-dimensional separation of concerns [1], permits the clean separation of multiple, potentially overlapping and interacting concerns simultaneously, with support for on-demand re-modularization to encapsulate new concerns at any time.

Modularization [2] is decomposition into modules. A module is a software unit with well-defined interfaces which express the services that are provided by the module for use by other modules. A module promotes information hiding by separating its interfaces from its implementation.

If the modeling technique is applicable to aspects, modularity of the composition specification is also of interest, that is, if it is separated from the aspects that are being composed.

How well does the modularity provided by the modeling technique support separation of concerns?

____

If the technique supports module composition (such as in an AO approach), how does it also support modularization of the composition specification?

____

If the modeling technique is subject, view, or aspect-oriented, concerns can be treated in different ways. Is the technique

☐ Symmetric ☐ Asymmetric

☐ Other: ____

Composability:

Composition is the act of creating new software units by reusing the existing ones. Three special sorts of compositions are distinguished: composition of crosscutting concerns, probabilistic compositions, and fuzzy compositions [3].

If a language provides mechanisms for modularization of crosscutting concerns, it must also provide operators for the compositions of these concerns with base concerns. Aspect weaving and superimposition are two examples of such operators. The composition of crosscutting concerns can be syntax-based or semantic-based [4]. In the former, the composition is based on syntactic references to base concerns. This may lead to the well-known fragile pointcut problem, where structural changes in the base concerns may invalidate the compositions. This problem is tackled in semantic-based composition by relying on the meaning of the relationships to be captured by the composition rather than the structure of the base concerns or specific naming conventions.

Probabilistic compositions are required if there are uncertainties in the specification of problems. Here, it may be preferable to define and optimize the solutions and the composition of solutions with respect to the probabilistic problem definitions [3].

Fuzzy compositions are required if a problem can be solved in a number of alternative ways, and it is not possible (or desired) to commit to a single alternative. Here, each alternative may be assigned to a fuzzy set that expresses the degree of relevancy of a solution. Special composition operators are required, which can reason about fuzzy sets.

What kinds of composition operators are supported by the technique?

☐ Explicit (e.g., weaving or superimposition) ☐ Probabilistic

☐ Fuzzy ☐ Syntax-based ☐ Semantics-based

☐ Other: ______________________________

☐ None

How does the technique support incremental composition?

____

If the modeling technique is aspect-oriented, composition can have additional properties. The following three questions apply to aspect-oriented techniques only.

How does the modeling technique support composition metrics?

____

What concepts does the technique support with respect to composition?

☐ Views ☐ Aspects

☐ Other: ______________________________

What properties are preserved in composition operators?

☐ Associativity ☐ Commutativity

☐ Other: ______________________________

Uniformity of models:

A language treats models uniformly, if it facilitates decomposing a model into a set of uniform and cooperating concerns, and it provides composition operators that manipulate such concerns in a uniform manner. The uniform treatment of the concerns helps to fulfil the closure property in the models. If a composition operator applied to any member of a set produces a model that is also in the set, then the set of models is closed under that operation [3]. The closure property is a way to standardize the decomposition mechanism and composition operators that are provided by a language. In addition, it helps to increase the abstraction level of concerns by forming a hierarchy of concerns in which the concerns at the higher levels of the hierarchy abstract from their lower level concerns.

Does the technique exhibit closure of its decomposition and composition mechanisms?

____

Traceability:

Traceability is (i) the ability to chronologically interrelate uniquely identifiable entities in a way that is verifiable or (ii) the ability to verify the history, location, or application of an entity (such as a concern, a process, an artifact) by means of documented recorded identification [5].

Traceability can be vertical or horizontal. Using requirements as an example, vertical traceability is the ability to trace requirements back and forth through the various layers of development, e.g. through the associated life-cycle work products of architecture specifications, detailed designs, code, unit test plans, integration test plans, system test plans, etc. It is possible to generalize this definition of vertical traceability to refer to abstraction levels above (e.g., system engineering or business process engineering) and below (e.g., architecture) the requirements [6].

Again using requirements as an example, horizontal traceability refers to the ability to trace requirements back and forth to associated plans such as the project plan, quality assurance plan, configuration management plan, risk management plan, etc. This definition can also be generalized. There can be horizontal traceability at the design level (tracing across different design documents) and at levels above requirements (system engineering or product family domain analysis).

It may be argued that traceability is the concern of tools, however, some entities may be explicit in the modeling approach that are, or could be, targeted for traceability.

What are the entities that should be traced either in the technique (i.e., as part of the process of applying the technique), or in models produced by the technique?

____

Trade-off analysis:

Trade-off analysis is defined as "Determining the effect of decreasing one or more key factors and simultaneously increasing one or more other key factors in a decision, design, or project" [7]. Trade-off analysis offers the ability to choose between a set of alternatives, based on a set of potentially conflicting criteria.

Does the modeling technique:

- ☐ Integrate trade-off analysis as part of the technique and/or tools
- ☐ Support trade-off analysis by providing easy access to relevant data
- ☐ Have no concept of trade-off analysis
- ☐ Other: ____

Specifying and analyzing quality attributes:

Quality attributes are non-functional requirements that are used as criteria to judge the quality of a system, rather than its specific behavior. Techniques to specify and measure such requirements are, e.g., performance specification using the UML MARTE profile followed by system design performance analysis.

Does the modeling technique allow specification and analysis of quality attributes?

____

Mapping to the next stage of software development:

Simply speaking, this is vertical mapping; how we go from one development phase to another.

How does the modeling technique address moving to the “next” stage of software development, whatever that might be?

Tool support:

Tools automate formalized parts of one or more activities within and across software development phases (e.g., requirements, design, validation, deployment, runtime management, evolution, etc.). There are at least two aspects of a tool that are important (from a classification perspective): its scope and its underlying “paradigm”. Tools are typically built to support a technique or method, that is, the technique or method underlying a tool determines the activities targeted by the tool (its scope) and the paradigm that the tool supports (i.e., the concepts, abstractions, and practices that constrain how the activities and associated development artifacts targeted by the tool are handled or represented). For example, in the OO paradigm, modules are classes/abstract data types, while in the procedural paradigm modules are functional abstractions.

What are the scope and paradigm of tools for the approach?

What are the scope and paradigm of tools for validating/verifying either the approach or models produced by it?

What are the scope and paradigm of tools for analyzing the consistency of either the approach or models produced by it?

AO techniques support separation of concerns along many dimensions, and thus it is important to provide tools that help modelers understand the relationships across the different views as they evolve and to analyze the relationships to identify problems arising from inconsistent representation of structural and behavioral elements and from undesirable interactions of behaviors across different views. Tools for visualizing relationships across the different views and for analyzing consistency and interactions are needed to make such techniques useful and scalable.

How do tools help modelers visualize relationships across different views of the approach?

How do tools help modelers analyze consistency and interactions for the approach?

Empirical studies/evaluation (only needs to be answered by the submitter of the model(s) accepted at the CMA workshop):

Empirical studies are classified as controlled experiments, case studies, and surveys [8]. A case study is an in-depth observational study, whose purpose is to examine projects or evaluate a system under study. Case studies are mostly used to demonstrate the application of an approach. A survey is an exploration of existing literature. It is a descriptive research method and provides no control over the measurements. A survey considers the precedent work in the broader sense [8]. A controlled experiment [8] is performed in an controlled environment with the aim to manipulate one or more variables and control other variables at fixed values to measure the effect of change.

What kinds of empirical studies or evaluations have been performed for the technique?

Expressiveness:

A modeling technique can be expressive enough to model domain-specific concepts, specific non-functional requirement concepts (e.g., resource allocation), more general concepts such as objects, etc. It can also be expressive in modeling relations among these concepts and in its manipulations of the concepts/relations (for example in expressing how they should be composed or transformed).

Describe the expressiveness of the modeling technique:

---

3. Additional Key Concepts (“parking lot”)

Noted below, there are several additional concepts that the original workgroup identified, but were unable to adequately discuss at the workshop to include in the previous descriptions. Please feel free to comment on a modeling technique’s capabilities in any of these areas if you would like, but be aware that discussions around these concepts may not be possible at the workshop.

Scalability:

Scalability evaluation is the process of assessing how the cost-effectiveness (which needs to be defined) of an approach evolves as a function of the complexity of the case studies it is applied to. For instance, complexity can be measured in terms of the size of models, which for example, can be measured in terms of number of modeling elements.

Abstraction:

Intermodule dependency and interaction:

Reusability:

E.g., parameterization, libraries, templates…

Usability:

Readability, understandability

Reduction of modeling effort:

Modeling effort can be measured in different ways. One way is to conduct a controlled experiment that compares the modeling effort of applying an AO approach with that of an OO approach. An alternate, much less expensive way, is to estimate modeling effort through a surrogate measure, such as the number of modeling elements required for the different approaches.

Evolvability:

Support for requirements changes

Completeness:

---

4. References

1. Harold Ossher and P. Tarr: “Multi-Dimensional Separation of Concerns and the Hyperspace Approach”. In Software Architectures and Component Technology, M. Aksit (Ed.), Kluwer Academic Publishers, pp. 293 - 323, 2002.
2. David Parnas: “On the Criteria To Be Used in Decomposing Systems into Modules”. In Communications of the ACM, vol. 15, pp. 1053—1058, 1972.
3. Mehmet Aksit: “The 7 C's for Creating Living Software: A Research Perspective for Quality-Oriented Software Engineering”. In Turkish Journal of Electrical Engineering and Computer Sciences, vol. 12, pp. 61--95, no. 2, 2004.
4. Ruzanna Chitchyan, Phil Greenwood, Americo Sampaio, Awais Rashid, Alessandro Garcia, and Lyrene Fernandes da Silva: “Semantic vs. syntactic compositions in aspect-oriented requirements engineering: an empirical study”. In Proceedings of the 8th ACM international conference on Aspect-oriented software development, pp. 149—160, 2009.
5. ISO 9001: 2000, 2009, Quality management systems — Requirements, http://www.iso.org/iso/catalogue_detail?csnumber=46486
6. Tim Kasse, Practical Insight into CMMI, Artech House Publishers, ISBN: 1580536255, pp 153-154, 2008.
7. Business Dictionary, http://www.businessdictionary.com/definition/tradeoff-analysis.html, 2011.
8. Claes Wohlin, Per Runeson, Martin Höst, Experimentation in Software Engineering: An Introduction, Springer 1999