Report on the 7th International Workshop on Models@run.time

Nelly Bencomo
INRIA Paris-Rocquencourt, France
nelly@acm.org

Gordon Blair
Lancester University, UK
gordon@comp.lancs.ac.uk

Sebastian Götz
Technische Universität Dresden, Germany
sebastian.goetz@acm.org

Brice Morin
Stiftelsen SINTEF, Norway
brice.morin@sintef.no

Bernhard Rumpe
RWTH Aachen, Germany
rumpe@se-rwth.de

ABSTRACT

The 7th edition of the workshop Models@run.time was held at the 15th International Conference on Model Driven Engineering Languages and Systems (MODELS). The workshop took place in the city of Innsbruck, Austria, on the 2nd of October 2012. The workshop was organised by Nelly Bencomo, Gordon Blair, Sebastian Götz, Brice Morin and Bernhard Rumpe. It was attended by at least 48 people. In this report we present a synopsis of the presentations and breakout discussions that took place during the workshop.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.9 [**Software Engineering**]: Management

General Terms

runtime models, runtime abstractions, runtime adaptation, reflection

Keywords

MDE, reflection, abstraction

1. INTRODUCTION

The Models@run.time workshop series provides a forum for exchange of ideas on use of runtime models to support self-* properties like self-awareness, runtime adaptation and self-healing. At least forty-five (48) people attended the 7th edition of the workshop. This included researchers from different research communities who work on model-driven software engineering, software architectures, computational reflection, adaptive systems, autonomic and self-healing systems, and requirements engineering. In response to the call for papers, eighteen (18) papers were submitted, of which four (4) long papers and seven (7) position papers were accepted and presented during the day. Each submitted paper was reviewed by at least 3 program committee members.

2. WORKSHOP FORMAT AND SESSION SUMMARIES

The activities during the workshop were structured into presentations and discussion sessions. In the opening presentation, Nelly Bencomo set the context of the workshop by summarizing the major results from previous workshop editions, announcing partial results of the Dagstuhl Seminar on the topics held in December 2011 [1], and outlining the path to follow during the workshop. The opening presentation was followed by the papers sessions with four (4) long presentations and seven (7) shorter presentations. The papers with long presentations where assigned a second reader, who was asked to prepare a set of questions for the respective paper beforehand.

All presentations were done during the morning to allow enough time for discussion during the second part of the day. In the afternoon, the workshop participants formed four (4) groups. Each group was charged with discussing the same set of questions under a relevant topic. At the end of the workshop, each group selected a representative who presented the questions raised during the discussions and also the conclusions reached by the group. More details about the discussion sessions can be found in section 3. The eleven (11) presentations were divided into four (4) paper sessions, which will be outlined as follows.

Session 1 : Requirements and Goals

This session focused on runtime models used in requirement engineering. In particular, the application of goal models at runtime expressed in the goal-oriented requirements language (GRL), the user requirements notation (URN) and the i-* modeling language have been discussed. This session covered the following two (2) long papers:

Goal Models as Run-time Entities in Context-Aware Systems. Mira Vrbaski, Gunter Mussbacher, Dorina Petriu and Daniel Amyot. Second Reader: Nelly Bencomo

Satisfying Requirements for Pervasive Service Compositions. Luca Cavallaro, Pete Sawyer, Daniel Sykes, Nelly Bencomo and Valerie Issarny Second Reader: Gang Huang

Session 2 : Self-adaptation

The aspect of adaptation in the models@run.time approach has been subject to this session. Three (3) position papers have been presented, which covered topics from continuous validation of adaptation, the adaptation of runtime models themselves to the application of models@run.time to adapt the feedback loop of the self-adaptive system (as usually only the system is subject to adaptation by the feedback loop).

A Process for Continuous Validation of Self-Adapting Component Based Systems. Viet-Hoa Nguyen, NoŚl Plouzeau, François Fouquet and Olivier Barais.

Model Execution Adaptation? Eric Cariou, Franck Barbier and Olivier Le Goaer.

Actor-based Runtime Model of Adaptable Feedback Control Loops. Filip Krikava, Philippe Collet and Robert France.

Session 3 : UML and DSLs

This session focused on the application of the unified modeling language (UML) and domain specific (modeling) languages

DOI: 10.1145/2413038.2413061 http://doi.acm.org/10.1145/2413038.2413061

[BBG+13] N. Bencomo, G. Blair, S. Götz, B. Morin, B. Rumpe
Report on the 7th International Workshop on Models@run.time
In: ACM SIGSOFT Software Engineering Notes, vol. 38, no. 1, pages 27-30. January 2013.
www.se-rwth.de/publications

(DSLs) to express models@run.time. One (1) long paper and three (3) position papers were presented, which covered the application and utilization of DSLs, UML and fUML (a subset of the UML enriched with semantics to allow for executable models), at runtime, as well as an approach to analyze structural consistency across models specified in different languages.

Model-Driven Development of i-DSML Execution Engines. Gustavo Sousa, Fabio Costa, Peter Clarke and Andrew Allen.
Second Reader: Frank Trollmann

Towards Supporting Multiple Execution Environments for UML / OCL Models at Runtime. Lars Hamann, Martin Gogolla and Daniel Horsel.

A Runtime Model for fUML. Tanja Mayerhofer, Philip Langer and Gerti Kappel.

Expressing Model Relations as Basis for Structural Consistency Analysis in Models@run.time. Frank Trollmann and Sahin Albayrak.

Session 4 : Applications of Models@run.time

This final presentation session covered applications of models@run.time approaches. One (1) long and one (1) position paper were presented. In particular the application of models@run.time for the development of graphical user interfaces for legacy systems and an idea to achieve service interoperability under an open world assumption were discussed.

Rapid GUI Development on Legacy Systems: An Runtime Model-Based Solution. Hui Song, Michael Gallagher and Siobhan Clarke.
Second Reader: Arnor Solberg

A semi-automatic behavioral mediation approach based on models@runtime. Runze Hao, Brice Morin and Arne Berre.

3. BREAKOUT-SESSIONS

The afternoon sessions of the workshop were used for breakout group discussions. 18 workshop participants and the five organizers formed four groups by interest.

The group topics were negotiated at the beginning of the first session after lunch and reflect the participants special interests w.r.t. models at runtime. The specific questions *what do we know so far?* and *what do we want to know?* were chosen as the questions to drive and motivate the group discussions. In the following the results of the discussions will be discussed groupwise.

3.1 Group A: Applications of models@run.time

A smaller group yet productive discussion group focussed on the currently already existing and also possible applications of M@RT. Bernhard Rumpe was the representative of this group and the text below is based on the presentation of the results of their discussions. It was highlighted how the identified applications demonstrate the potential industrial relevance of M@RT-technologies and give impulses to the foundational questions of models@run.time. It was concluded that definitely, there is already a number of applications using models@run.time in a variety of forms in industrial use. However, many of the applications so far have used proprietary approaches as we just now come up with more systematic uses of models@run.time.

There are many aspects that models@run.time can serve, but predominantly *flexibility* and *adaptability* need to be mentioned. It, however, depends on the domain and the application what needs to become flexible. Therefore there will not be a single killer application of models@run.time.

Some facts that we know about models@run.time from an application's point of view are:

- models@run.time should use models to make certain aspects of a system flexible and adaptable. Flexibility depends on the kind of system its uses. Thus, we have many different kinds of models and purposes at runtime.
- models@run.time is fairly new and therefore it's ideas not very much used yet in industrial strength applications. It is currently the other way round: models@run.time-research is being fostered by existing industrial applications, that solved their flexibility-problems through individual solutions.

What we don't know so well yet:

- What's the differences between flexibility, evolution and variability? It can be assumed that models@run.time can be seen as a form to provide variability of a software product line that is being fixed very late – namely at runtime – either by the system itself, by the user or by some experienced on-site-available developer to customize a system.
- How these fit together in a sound and robust way? May be we can assume that software product line techniques can be expanded to models@run.time techniques at runtime. However, models@run.time is more than only selection of the best configuration, as adapting a model allows so many more choices of adaptation than just selection of given alternatives.
- From the economic side: Will models become useful assets on their own now, as they might be present at runtime and thus present for the user? May be models become much better reusable, if not tradable assets?

Some key questions for the models@run.time community are definitely:

- How to ensure quality for a model used and changes at runtime? How to test quality of such a runtime configurable system?
- Is models@run.time applicable to software within physical / organisational systems other than pure software? There is a strong assumption that yes, models@run.time will be very useful in domains, where models about the controlled environment are of use.

The working group also looked at a possible list of current M@RT applications (and found a number of systems with widely varying adaptivity). Some key examples are:

- data base management systems (DBMS) know nothing about their data structures, but are completely configurable
- protocols, web management systems etc. are also fully generic and thus configured. These configurations are usually no explicit models, but serve a similar purpose.

DOI: 10.1145/2413038.2413061 http://doi.acm.org/10.1145/2413038.2413061

- SAP is an excellent example for a highly runtime configured system: both data structures as well as workflows are not part of the SAP code itself, but added through explicit configuration. Entity-Relationship-Models as well as ARIS-workflow models are used for this purpose.

Of course there are many possible future applications. The group only discussed two as exemplary for a variety of models: Design models, e.g. for buildings/city infrastructure to create management models for the real physical system; and operative (workflow) models to monitor/guide activities e.g. construction, production, and bureaucracy.

3.2 Group B: What are models@run.time Systems?

This group was led by Eric Cariou and focused on the question, what models@run.time systems are and how they align with model execution. This topic is reflected in the papers presented at the workshop, which covered approaches, which clearly adhere to the current understanding of models@run.time, but also approaches which focus on runtime models as executable models. The aim of this group was to identify how executable models fit into the models@run.time paradigm.

Hence, first, a general definition of a model@run.time was discussed. According to the group, models@run.time is an abstraction of a running system that is being manipulated at runtime for a specific purpose. A model is by definition an abstraction. In particular, the model reflects a running phenomena in a given environment. The key aspects of model manipulation are model observation and model modification. Based on these two main kinds of model manipulation, the focus of models@run.time systems is notably on adaptation, monitoring, trace of execution or debugging.

Next, model execution has been characterized. Here, the running system is an execution engine interpreting a model. In analogy to the models@run.time idea, runtime model manipulation is performed by modifying the model for a specific purpose of direct model execution. Still following the models@run.time definition, the model is an abstraction of the running system but here in a tricky way. Indeed, the model is the abstraction of the composition of itself and of the execution engine interpreting it. The group proposed that model execution can be considered as a kind of models@run.time even if it remains a particular vision of them. Having a model execution as a runtime models created debate as it was not obvious for some of the attendees.

3.3 Group C: Verification, Validation and Unanticipated Evolution

Two key constituents of models@run.time systems, which are strongly intertwined, are the reasoning part and the adaptation part. In terms of a feedback loop underlying models@run.time systems, these represent the plan/decide and act/execute phases. This group has focused on these topics and, especially, on validation and verification aspects.

First, the group had a general discussion on what is known from a reasoning point of view on models@run.time systems. First, the performance of models@run.time system has been identified to be two-sided. At runtime we may know that we just need to verify a small part, whereas at design time we need to check all possible system variants. Then, this two-sided nature has been discussed w.r.t. verification. On the one hand, the system is evolved by the reasoner, which ensures a consistent new configuration. In this case, V&V is only required to ensure that no unexpected changes to the system occured. On the other hand, the system could evolve by itself in an unanticipated manner, where the feedback loop performs corrective actions. Here, we need to check if the appeared configuration is valid in any case. Next, the discussion covered the question how to handle open systems (i.e., models@run.time under open world assumption). The group concluded, that it is important to define boundaries for runtime adaptation to keep the system verifiable/manageable. This is because the boundary limits the variability space to be handled. Of course, this violates the open world assumption, but is still more open, than the closed world assumption. Thus, the group discussed how open models@run.time systems should be. Notably, these boundaries are indicated by connections of models and by metamodels. This idea can be paraphrased as constrained evolution to keep the system verifiable. Another topic, which emerged in the discussion was that also for models@run.time systems, there are problem space models and solution space models and the insight that we don't verify in problem space. Finally, the discussion focused on the correctness of the models used at runtime. The group agreed, that in general, we cannot be sure about the correctness of our models collected at runtime. But, at least we can collect data and make statistical statements about their correctness.

Next, the group discussed what is not known, yet, but the community strives for answers. As a result, a set of questions have been collected:

- Are V&V tools at runtime performant enough?
- Are V&V tools able to provide meaningful results (due to too strong assumptions) ?
- How open do we need to be?
- How should a development processes for M@RT Systems look like?

Finally, a set of questions the group thought should be addressed in next years edition of the models at runtime workshop have been collected:

- Work on real applications!
- Methodologies / Which tools/approaches are best for which purposes?
- Meta-Reasoning / M@RT-Hierarchies

3.4 Group D: Adaptation and Reasoning

Frank Trollmann was the representative of the group that discussed "Adaptation & Reasoning". He started by subsuming the groups discussion regarding"What we know". The opinion of the group is that a variety of approaches for adaptation and reasoning already exists in the state of the art. These approaches can be found inside of the models@run.time community as well as in conventional Model-Driven Engineering and other domains. During the discussion of the group, it was noted that existing techniques from the fields of artificial intelligence and formal specification can be applied. Architecture models are a close example to the domain of models@run.time. They have been broadly adapted and used as basis for reasoning about the running system in several approaches.

Frank continued to discuss the group's result regarding "What we don't know". One of the main questions in this area is how to

DOI: 10.1145/2413038.2413061 http://doi.acm.org/10.1145/2413038.2413061

structure and apply existing approaches at runtime. This concerns the fulfillment of runtime requirements like performance and memory-efficiency as well as conceptual problems like the integration of new analysis and adaptation methods into existing models@run.time approaches.

Another point of discussion was the management of runtime models. While several existing methods deal with the adaptation of models to cause adaptations of the running application, the area of detecting such adaptations and adapting the models to reflect them correctly can be considerably extended. In this context variable approaches are of special complexity. Such approaches are able to learn the behavior of a running system and construct or adapt models to reflect this learned knowledge. This can even result in models conceived at runtime that reflect dynamic aspects of the application or its context-of-use which cannot be foreseen at design time. As an example, an application could learn reoccurring interaction behavior of a user or even new requirements or tradeoffs between soft-goals that can occur in a special situation. Correctly conceiving models and keeping them consistent with each other and with their system under study is a foundation for successful reasoning as the quality of the reasoning results depends on the quality of the input models. Nelly Bencomo supplemented these points with the remark that the synchronization of the requirements and goals of the model and the actual system is especially vital in order to assure that models are correctly adapted to serve the fulfillment of the systems goals.

Frank closed with the remark that while the group attempted to structure existing work they were sidetracked by a discussion related to the topic "What are models at runtime". The group came to the conclusion that the variety of existing approaches requires a clear definition of models@run.time. This definition should be usable to structure these approaches such that a more rigorous analysis on adaptation and reasoning and the application of existing techniques can be made. The groups opinion regarding the key-question to be answered in future editions of the workshop was thus "What are models@run.time?". As a first step towards a categorization the group identified the problem space addressed by a model to be a promising approach for categorization as the information reflected in the models and derivable from reasoning algorithms directly depends on what information is needed for a solution in this problem space.

A general wrap-up discussion was held at the very end of the workshop day. The workshop was closed with a warm "thank you" from the organizers to all participants for another successful workshop.

4. CONCLUSION

Currently, some of the central challenges in software engineering are about the need to develop software models and methods for building software systems that (i) can dynamically adapt to context and environment, (ii) can assemble themselves on-the-fly and (iii) are aware of themselves and offer account for themselves (by self-explanation). We anticipate that models@run.time and the required reflective capabilities will play an integral role to tackle these challenges. Several of the papers presented in this edition and earlier editions of the workshop have tackled or acknowledged those challenges. Thus, we think more research effort needs to be done in order to come up with substantial results in this context. We also foresee future editions of successful workshops and conferences tracks in this, highly interdisciplinary topic, bringing together researchers from software engineering, artificial intelligence, control engineering and many more.

5. REFERENCES

[1] Assmann, U., Bencomo, N., Cheng, B. H. C., and France, R. B. Models@run.time (dagstuhl seminar 11481). *Dagstuhl Reports 1*, 11 (2011), 91–123.

Acknowledgments

No workshop is successful by the efforts of only a few people. We would like to thank the members of the program committee who acted as anonymous reviewers and provided valuable feedback to the authors: Thais Vasconcelos Batista, Walter Cazzola Franck Chauvel, Peter J. Clarke, Siobhan Clarke, Fabio Costa, Kerstin Eder, Holger Giese, Martin Gogolla, Lars Grunske, Gang Huang, Jean-Marc Jézéquel, Marin Litoiu, Hausi A. Muller, Rui Silva Moreira, Arnor Solberg, Hui Song, Matthias Tichy, Mario Trapp. Special thanks to Eric Cariou and Franck Trollmann for the support while writing this summary.