Summary of the 5th International Workshop on Models@run.time

Nelly Bencomo[1] Gordon Blair[1]
Franck Fleurey[2] Cédric Jeanneret[3]

[1] Computing Department, Lancaster University, UK
[2] SINTEF, Oslo, Norway
[3] Department of Informatics, University of Zurich, Switzerland
nelly@acm.org, gordon@comp.lancs.ac.uk,
Franck.Fleurey@sintef.no, jeanneret@ifi.uzh.ch

Abstract. The 5th edition of the workshop Models@run.time was held at the 13th International Conference MODELS. The workshop took place in the exciting city of Oslo, Norway, on the 5th of October 2010. The workshop was organised by Nelly Bencomo, Gordon Blair, Franck Fleurey, and Cédric Jeanneret. It was attended by at least 33 people from more than 11 countries. In this summary we present a synopsis of the presentations and discussions that took place during the workshop. **Keywords:** runtime adaptation, MDE, reflection, run-time abstractions.

1 Introduction

The Models@run.time workshop series provides a forum for exchange of ideas on the use of run-time models. The workshop series targets researchers from different communities, including model-driven software engineering, software architectures, computational reflection, adaptive systems, autonomic and self-healing systems, and requirements engineering. This edition of the workshop successfully brought together researchers from different communities and, at least, thirty three (33) people from eleven (11) countries attended the workshop.

In response to the call for papers, fifteen (15) papers were submitted, of which four (4) papers and six (6) posters were accepted. Every submitted paper was reviewed by at least 3 program committee members. The papers presented during the workshop are published in a workshop proceeding [1]. Two papers were selected as the best papers. Extended and improved versions of these two papers are published in this post workshop proceedings with other selected papers from all the workshops at MODELS 2010.

2 Workshop Format and Session Summaries

The workshop activities were structured into presentations, posters, and discussion sessions. In the opening presentation, Nelly Bencomo and Franck Fleurey set the

context of the workshop by summarizing the major results from past workshop editions and outlining the path to follow during the workshop. The opening presentation was followed by the papers and posters sessions.

In the paper sessions four (4) papers were presented. Authors presented their papers in a twenty-minute-time slot, and five minutes were allowed for questions and discussion. *Robert France and Arnor Solberg* chaired these presentations. In the poster session, six (6) authors also presented their work to the workshop attendees.

Ppaper and poster 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 three groups. Each group took care of discussing specific relevant topics. At the end of the workshop, each group selected a delegate who presented the conclusions and research questions raised by the group. More details about the discussion session can be found in Section 3. The four (4) paper presentations and the six (6) posters were divided into the following two sessions:

Session 1: Fundamental Concepts

- *Meta-Modeling Runtime Models*, by Grzegorz Lehmann, Marco Blumendorf, Frank Trollman and Sahin Albayrak.

- *Toward Megamodels at Runtime*, by Thomas Vogel, Andreas Seibel and Holger Giese.

Session 2: Evaluation and Experimentation

- *Applying MDE Tools at Runtime: Experiments upon Runtime Models*, by Hui Song, Gang Huang, Franck Chauvel and Yanchun Sun.

- *Run-Time Evolution through Explicit Meta-Objects*, by Jorge Ressia, Lukas Renggli, Tudor Girba and Oscar Nierstrasz.

Session 3: Applications

The following posters were displayed, presented and discussed with the workshop attendees.

- *A Model-Driven Approach to Graphical User Interface Runtime Adaptation*, by Javier Criado, Cristina Vicente-Chicote, Nicolás Padilla and Luis Iribarne
- *Monitoring Model Specifications in Program Code Patterns*, by Moritz Balz, Michael Striewe and Michael Goedicke
- *Separating Local and Global Aspects of Runtime Model Reconfiguration*, by Frank Trollmann, Grzegorz Lehmann and Sahin Albayrak.
- *Using Models at Runtime For Monitoring and Adaptation of Networked Physical Devices: Example of a Flexible Manufacturing System*, by Mathieu Vallee, Munir Merdan and Thomas Moser.
- *Monitoring Executions on Reconfigurable Hardware at Model Level*, by Tobias Schwalb, Graf Philipp and Klaus D. Müller-Glaser.
- .*Knowledge-based Runtime Failure Detection for Industrial Automation Systems*, by Martin Melik-Merkumians, Thomas Moser, Alex*ander Schatten, Alois Zoitl and Stefan Biffl.*

3 Discussions

During the afternoon, three discussions groups were established. Each group was asked to discuss *a* set of topics based on the questions raised during the presentations and the research interests of the attendees. The following are the main topics discussed during the afternoon:

Topic of discussion group 1:

- *Classification and applications of models@run.time*

Topics of discussion group 2:

- *Abstraction gap*
- *Causal connection.*
- *Temporal Gap.*

Topic of discussion group 3:

- *Difference between design time and run-time models*

After spending two hours debating the presentations and shared research interests, each group presented a summary of their discussions and conclusions.

Thomas Moser was the representative of the breakout discussion group with the topic "*classification and applications of models@run.time*". The discussions of the group started based on the figure “Categories of run-time models” from the paper by Thomas Vogel et al. (Fig 1, page 15 in [1]). Thomas Vogel et al. identified the following types of models: implementation models, configuration and architectural models, configuration space and variability models, context and resource models, as well as rules, strategies, constraints, requirements and goals. Based on this figure, the group defined reasoning (analysis) and decision as the two major purposes of models@run.time. Examples for reasoning could be faults and defects detection and model checking; examples for decisions could be changes to running systems such as runtime adaptation or the generation of reports about behaviour.

Using these two major purposes for models@run.time, the group developed a table-based structure showing which types of models are used and for which purposes, i.e., for which type of reasoning and for which type of decision. The columns of the table can also be considered as the dimensions of models@run.time: namely type of model, type of reasoning and type of decision. Thomas commented that a fourth dimension could also be considered if context is included.

Mathieu Vallee was the representative of the group with the topic "*causal connection and abstraction gap*". The causal connection, i.e., how a running system is reflected in a runtime model (and conversely), has been discussed in previous editions of the workshop. However, this year participants focused on the difficulties that arise when the abstraction gap between a runtime model and the modelled system expands. Typically, as systems become more complex (larger, and distributed), it looks relevant the use of models that are more abstract and therefore easier to understand. However, and at the same time, this makes the causal connection more difficult to establish and to maintain. A lengthy discussion took place to clarify the problems, and is summarized as follows:

- In state-of-the-art solutions using runtime models, the abstraction gap is kept small, so that the relation between the system and the models is relatively easy to establish. A typical example is the use of a runtime architectural model together with a component-based infrastructure, in which concepts in the model (e.g., component) directly represent elements in the system.
- In some works, the abstraction gap becomes broader due to the adoption of higher level models. For instance, research on requirements at runtime raises questions on how to establish a causal link between a model of requirements and components in an underlying system.
- In other works, the abstraction gap may become broader due to the manipulation of the system at a finer level of granularity. For instance, using meta-objects at runtime enables the manipulation of a system at a fine granularity, but it was not clear to the group how it can be linked to an architectural model.

Similar issues arise regarding the "*temporal gap*", which appears when the model and the system cannot be synchronized anymore or at least temporarily, i.e., the system still evolves while building the runtime model. In some pure software systems (i.e. no hardware is taken into account), it is possible to keep the model and the system always synchronized (e.g. by freezing execution while building the model). In systems involving physical components, as well as in more complex software systems, continuously maintaining the synchronization may not be possible. Mathieu emphasized that we need to take into account that a runtime model may not always accurately reflect the current system state, and that we need to design methods for estimating the temporal gap, as well as compensating it (e.g. through prediction of future states).From the discussions in the group, it appears that addressing these issues is a rather long-term objective. Nevertheless, the group believe that difficulties arise more due to the purpose of a runtime model than from its level of abstraction. As a consequence, the group recommended the following steps:

1. Elaboration of a classification of runtime models, according to their purpose.
2. Elaboration of concrete examples involving several models/levels of abstraction.
3. Study of the relationship between different runtime models in a given system.
4. Study of general solutions for managing models with different levels of abstraction.The last presentation was given by *Betty Cheng* who was the representative of the group that discussed "*differences between design time and run-time models*". The discussions of this topic focused on the identification of requirements for run-time models. Betty reported that the group discussed about the contents of a run-time model (in contrast to a design model). According to Betty and colleagues, run-time model comprises:

- Environmental conditions depicted by design info (e.g., plant model) and run-time info (e.g., plant model with values). Run-time info would be more abstract.
- System conditions depicted by design info (traditional models like class, state, etc.) and run-time info (e.g., current task, service, attributes, processing node). Run-time info would also contain traceability information to design-time info.

The group also discussed about the purposes of run-time models and specifically focused on the case of self-adaptive systems. The following were the purposes identified:

- Monitoring the state of the system and the environment
- Decision-making: to process data to adapt; validation and simulation are also included
- Adaptation (mode change; reconfiguration). The changes can affect the structure and behaviour of the system.

The group presented the following recommendations:

- Move towards multiple run-time models, rather than using a monolithic run-time model
- Take into account the purpose for run-time models. The kinds of run-time models strongly depend on what we want to do with the system (e.g. performance analysis; fault tolerance, diagnosis; adaptive; safety)
- Identify possible purposes of run-time models and find additional ones (change existing model types, and consider to develop new ones)

Final Remarks: It is interesting to note that discussions of the different groups converged to similar outcomes and recommendations; as for example the need of a classification for run-time models and identification of the purposes and relationships between the models. A general wrap-up discussion was held at the very end of the afternoon. The workshop was closed with a friendly "thank you" from the organizers to all participants for a fruitful workshop. After the workshop, the organizers used the feedback from attendees and PC members to select the best 2 papers. After careful discussion, the best papers were selected and are presented in this book.

Acknowledgments. No workshop is successful by the efforts of only a few people. *We would al*so like to thank the members of the program committee who acted as anonymous reviewers and provided valuable feedback to the authors: *Uwe Assman, Franck Chauvel, Betty Cheng, Peter J. Clark, Fabio Costa, Jeff Gray, Holger Giese, Oystein Haugen, Jozef Hooman, Gang Huang, Paola Inverardi, Jean-Marc Jezequel, Rui Silva Moreira, Flavio Oquendo, Arnor Solberg, Mario Trapp, and Thaís Vasconcelos Batista*. We thank Mathieu Vallee and Thomas Moser in the elaboration of Section 3 of this summary. Last but not least, we thank the authors for their interesting submissions and for helping us making this workshop possible.

References

[1] Nelly Bencomo, Gordon S. Blair, Franck Fleury, and Cédric Jeanneret, editors. *Proceedings of the 5th Workshop on Models@run.time, held at the ACM/IEEE 13th International Conference on Model Driven Engineering Languages and Systems (MODELS'2010)*, volume 641 of *CEUR Workshop Proceedigns*. CEUR, 2010.