2nd International Workshop on Models@run.time

Nelly Bencomo[1], Robert France,[2] and Gordon Blair[1]

[1] Computing Department, Lancaster University., InfoLab21,
Lancaster, UK, LA1 4WA,
[2] Computer Science Department, Colorado State University
Fort Collins, CO, USA, 80523-1873
{nelly, gordon}comp.lancs.ac.uk, france@cs.colostate.edu

Abstract. The second edition of the workshop Models@run.time was co-located with the ACM/IEEE 10th International Conference on Model Driven Engineering Languages and Systems. The workshop took place in the lively city of Nashville, USA, on the 2nd of October, 2007. The workshop was organised by Nelly Bencomo, Robert France, and Gordon Blair and was attended by at least 25 people from 7 countries. This summary gives an overview of the presentations and lively discussions that took place during the workshop.

Keywords: model-driven engineering, reflection, run-time systems.

1 Introduction

Research on model-driven software development has mainly focused on the use of models at design, implementation, and deployment stages of development. This work has produced relatively mature techniques and tools that are currently being used in academia and industry. However, the use of model-driven approaches for validating and monitoring run-time behaviour can also produce significant benefits. A key benefit is that models can be used to provide a richer semantic base for runtime decision-making related to system adaptation and other runtime concerns. For example, models can be used to help determine when a system should move from a consistent architecture to another consistent architecture. Model-based monitoring and management of executing systems can also play a significant role as we move towards implementing the key self-* properties associated with autonomic computing (i.e. self-management, self-optimization, self-healing, and self-protection)[3]

Goal

The goal of this workshop was to understand the relationship between models produced during development and models used to support and enable run-time monitoring and adaptation of software. Key research topics of the workshop were (1) to study how models produced during development can be effectively leveraged during run-time, (2) how model-driven approaches can be applied to managing and monitoring the execution and operation of systems, (3) to what extent can model-

driven engineering be used to tame the complexity of developing and managing adaptive software. This is the second in a series of MODELS workshops on this topic. The workshop successfully brought together researchers from different communities including researchers working on model-driven software engineering, software architectures, computational reflection, adaptive systems, autonomic and self-healing systems, and requirements engineering. At least twenty-five people attended from France, Germany, Norway, South Korea, The Nertherlands, UK and the US.

The call for papers invited submissions on a number of focus topics including: relevance and suitability of different model-driven approaches to monitoring and managing systems during run-time, compatibility (or tension) between different model-driven approaches, the role of reflection in maintaining the causal connection between models and run-time systems, experience related to the use of run-time models to adapt software systems, and the management and modelling of runtime variability using models.

In response to the call for papers, nine (9) papers were submitted, of which (6) papers were accepted. Each submitted paper was reviewed by at least 3 program committee members. After discussions two papers were selected as the best papers. The decision took into account the relevance of the papers to the goals of the workshop, the impact on the discussion and results, and the quality of the papers and presentations. These two papers were extended and improved taking into account the discussions and conclusions of the workshop and are published in this proceeding.

2 Workshop Format

The workshop was designed to facilitate focused discussion on the use of models during run time. It was structured into presentation and work (discussion) sessions. The opening presentation was given by Nelly Bencomo and Robert France. Nelly set the context of the workshop describing the general goal and presented the results of the 1st edition of the workshop in MODELS'06 [2] and the related workshop M-ADAPT (Model-driven Software Adaptation) at ECOOP'07 [1]. Robert continued by describing the specific goals of the second edition of the workshop and stating key questions to kick off the discussion and call for the inspiration and motivation needed during the rest of the day. After the opening presentation, the paper sessions followed. There were 6 papers divided in 3 sessions.

The workshop was structured into presentations during the morning and discussion sessions in the afternoon. During the presentation session, papers were presented by two speakers, the first speaker was an author of the paper and the second speaker (reader) was an independent reader. Second readers provided another view on the contents of the paper, placing it in relation to the workshop topics and research questions.

To ensure effectiveness of the format of the workshop, presentations were limited to 25 minutes, 15 minutes presentation by the first speaker, 5 minutes by the second reader and 5 minutes for questions. Presentation sessions were cochaired by Oystein Haugen and Arnor Solberg. After the presentations, specific research interests and questions were discussed. The partial results of this discussion were used to split the

participants into two groups to allow focused debate and dialogue during the afternoon. The workshop was closed by a final discussion, including an evaluation of the workshop itself made by the attendees. Details of the sessions are provided in Sections 3 and 4 below. The workshop proceeded smoothly, with all attendees keenly contributing through constructive and friendly debate. Attendees enjoy and praised the idea of the second readers.

3 Session Summaries

The 6 papers were divided into the following three categories according to their topics and contributions:

Error detection and Self Healing
- "*A Modeling Framework for Self-Healing Software Systems*", by Michael Jiang, Jing Zhang, David Raymer, and John Strassner, second reader: Jules White
- "*Model-Based Run-Time Error Detection*", by Jozef Hooman, and Teun Hendriks, second reader: James Hill

Monitoring and Verification
- "*System Monitoring using Constraint Checking as part of Model Based System Management*", by Christian Hein, Tom Ritter, and Michael Wagner, second reader: Matthias Gutheil
- "*AMOEBA-RT: Run-Time Verification of Adaptive Software*", by Ji Zhang, Betty Cheng, and Heather Goldsby, second reader: Jozef Hooman

Techniques and Approaches
- "*Coherent Support for Models at Run-Time through Orthogonal Classification*", by Atkinson Colin and Matthias Gutheil, second reader: Olivier Barais
- "*Control-theory and models at runtime*", by Pierre-Alain Muller and Olivier Barais, second reader: Aniruddha Gokhale

Robert and Nelly took notes of specific questions and topics raised during the presentations. After the presentations, individual questions made by the participants were gathered. Following the questions stated by the attendees and the notes taken by Robert and Nelly, discussions groups were established. Attendees left the room to start the exchange of ideas over meals

4 Discussions

After lunch each group came into the room and shared ninety minutes of lively discussions. Both groups addressed different topics. The first group discussed the *infrastructure* needed to support *the use of models during runtime*. The second group focused on *theoretical concepts and languages needed to define and work with models during runtime*. Slides summarizing the reports were produced by the leader

discussant of each breakout session (Oystein Haugen and Jules White respectively). As the two breakout groups reassembled to summarize their work it is worthy of note that the ideas of the paper “Control theory and models at runtime” by Muller and Barais had influence on both discussion groups.

Summary of discussion in the “Infrastructure Group”

The discussions in this group focused on what support is needed to effectively support the use of models during runtime. As a first step, the ways in which models were used in the papers presented at the workshop were discussed. The papers presented at the workshop mainly focused on *verification* (Zhang et al., Hein et al.), *error detection and correction* (Hooman et al.), and *self-healing* (Jiang et al.). In this context, runtime models specify the expected behaviours of the running application. These techniques use a "supervisor component" which observes the system in order to detect any deviation from its expected behaviour. Any deviation reveals a failure in the running application. If the objective is verification or error detection, the failure is reported to the user. In the case of self-healing, the supervisor automatically adapts the running application.

Philippe Lahire, one of the participants, suggested that a model@runtime is a model that is coupled with a controller that uses information from a running system to perform some control on the running system. The controller can be viewed as a separate model or can be considered to be part of the runtime model. The functionality provided by a controller is linked to the type of information captured in the runtime model, which, in turn, is tied to the type of adaptation supported by the model. He suggested that it would be useful to classify the properties that a runtime model must have to support particular types of adaptations (e.g., self-healing, self-protection) in the context of particular types of systems (e.g., information, embedded, distributed systems).

The discussions also led to the identification of other uses of runtime models. For example, runtime models can be used to support adaptation to environment changes, runtime updating of system components, or the graceful degradation of some functionalities. For each kind of adaptation, specific runtime models have to be defined. For example, in the case of adaptation to a changing environment, the models have to capture the variation points of the application and define rules to choose the appropriate variants according to the information coming from the environment. In the case of runtime updates of components, the model could capture the dependencies between running components and rules to safely upgrade them.

Given that there are various purposes for models at runtime (and corresponding types of models), the discussions then focused on identifying a common infrastructure to support the use of runtime models. The papers presented at the workshop (and the examples that came up in the discussions) all included a "supervisor component" around the runtime model. This component is called "runtime awareness" in the work of Hooman et al. and identified as the Observer/Controller pair in the control theory analogy proposed by Muller et al.. The supervisor component monitors data coming from the running application, makes decisions based on the observations and performs required adaptations on the running application. The supervisor thus consists of three activities: *Observe*, *Decide*, and *Adapt*.

For observation, the supervisor needs access to data coming from the running system. These data can be the inputs and outputs of the system but access to system internal data or environment information might also be required. For decision-making, the supervisor processes the information collected by observation using a decision model. This model defines appropriate responses for specific behaviours of the system. If needed, the decision can either lead to a user notification or trigger an adaptation of the running application.

Adaptation in this context, consists of changing the running system at runtime. It is not required for all systems and it cannot always be automated. In some cases, for example, error detection, debugging and verification, the user is notified and is responsible for modifying the system.

Summary of discussion in the "Concepts and Languages Group"

To start their discussions, this group considered it was pertinent to define what a model at runtime means. The group came up with the following definition: "*an abstraction at a higher level of representation than code that is used to derive adaptation through monitoring and feedback mechanisms*."

Using this concept, several use cases for models were identified.

- Model-based recovery
- System management
- Models for feedback
- Models for manipulating variability
- Determining what instrumentation to generate
- Models as an interface to a system rather than code manipulation

The group also concluded that current technologies like interceptors, AOP, constraint logic programming, control theory, and model checking techniques are useful to support the use of runtime models. This means that there is no need to come with new technologies in a short term.

Reflection was, as in the first edition of this workshop, stated as a capability needed when dealing with models to monitor and drive the execution of systems. In this sense, runtime models would be used to manipulate the system itself using introspective and intercession capabilities. It was discussed that the concept of meta-level would perhaps be interpreted in a different way in the context of models at runtime. Until now, meta-programming and reflection have been studied mainly at the coding level. At this point an open research question arised: what would be the parallel at the modelling level? The comparison would seem to be allowing the meta-model to change at runtime in order to adapt to new requirements or quality of service specifications. The above would require the meta-model to be continuously evaluated and understood during execution. This was contrasted with the current approach where a meta-model is interpreted somehow statically with no updates once the system starts. It was also pointed out that performance is an issue to take into account when using reflection. The ideas expressed by Pierre-Alain and Olivier paper about control theory seems to fit nicely with what was discussed

As the two breakout groups reassembled to summarize their work and exchange reached conclusions, it was interesting to see how both groups identified dynamic

software adaptation as a significant research area where the use of models at runtime can be useful. In this context, models can be used to (i) manipulate the system itself, e.g., adding a component to the model has the effect of changing the underlying code; and (ii) verify the behaviour of the system during execution (using for example reflection).

Final Remarks

A general wrap-up discussion was held at the end of the workshop. The organizers asked for feedback on the workshop and a number of useful ideas were suggested. Attendees confirmed that they were very pleased with the papers, presentations and discussions carried out during the workshop. The inclusion of second readers was considered successful and useful. It was concluded that the research community should be encouraged to continue the study the issues related to models at runtime and its relevance for the development of seld-adaptive systems. It was suggested that the organizers should consider the presentation of small demos that show the use of models at runtime in a possible next edition of this workshop. The workshop was closed with a warm "thank you" from the organizers to all participants for a successful workshop. After the workshop finished many of the attendees went for a well deserve dinner to continue talking.

Acknowledgments. We would also like to thank the members of the program committee who acted as anonymous reviewers and provided valuable feedback to the authors: Betty Cheng, Fabio M. Costa, Federal University of Goias, Brazil, Van Den Berg Aswin, John C. Georgas, Gang Huang, P.F.Linington, Andrey Nechypurenko, Eugenio Scalise, Rui Silva Moreira, Arnor Solberg, Marten van Sinderen, Thaís Vasconcelos Batista, Jules White. We also would like to thank Aniruddha Gokhale, Oystein.Haugen, Jim Hill. We specially thank to Philippe Lahire, Jeff Gray, Franck Fleurey, and Heather Goldsby for the feedback and ideas provided for this summary. Last but not least, the authors of all submitted papers are thanked for helping us making this workshop possible.

References

1. Blair, G., Bencomo, N., France, R., Cebulla, M.: Proceedings of the First Workshop on Model-driven Adaptation (M-ADAPT '07) at ECOOP 2007, Bericht Nr. 2007 - 10, (2007)
2. Blair, G., Bencomo, N., France, R.: Summary of the Workshop Models@run.time at MoDELS 2006. Lecture Notes in Computer Science, Satellite Events at the MoDELS 2006 Conference, Springer-Verlag (2006)
3. Kephart Jeffrey O., Chess David M. The Vision of Autonomic Computing, IEEE Computer, IEEE Computer Society Press, 41-50, (2003)