Preface

This book contains revised selected papers of six annual International Workshops on Behaviour Modelling - Foundations and Applications, which were held from 2009 to 2014 [1–6].

Behaviour modelling is about describing a system in terms of its states and its transitions from one state to another. The transitions may be initiated by the system itself or by the systems environment, including the system's users and other systems. Behaviour models capture system requirements forming the basis of their precision and completeness. They may have execution semantics, and, therefore, they are related to system simulation techniques.

Over the last decade, models have played an ever increasing role in software engineering, driven and dominated by the UML (Unified Modeling Language) and the MDA (Model Driven Architecture) standards. Today, it is possible that major parts of software systems can be generated from models fully automatically.

However, most of today's code generators work mainly from class diagram models and only address structural parts of the software. The code that implements behavioural business rules and governs the protocols of interaction between the user and the environment has to be added using traditional coding techniques. It seems that we have not been as successful in leveraging the power of modelling to raise the level of abstraction and eliminate low level coding in the behavioural realm as we have in the structural realm. Nor have we managed to properly exploit the potential of behaviour models to enable formal analysis and reasoning about behaviour, and thereby assure the quality of the final software.

This was the starting point of the series of workshops on Behaviour Modelling - Foundations and Applications (BM-FA). The objective was to raise awareness of the unrealised potential of behaviour modelling, to identify factors that have contributed to this, and ultimately to propose new ideas that overcome these barriers. We wanted to find better ways for modelling behaviour, for speeding up the software development process, better understanding its behaviour and guaranteeing its correctness.

This volume gives an overview of the ideas, problems, and solutions that were presented and discussed over the course of these six BM-FA workshops. The workshop papers and discussions explored the philosophy and practice of modelling, described the experience and problems with existing notations, and proposed new concepts for modelling behaviour and for combining existing modelling languages in new ways.

Modelling Practices

This volume starts with the paper written by Haim Kilov, a former academic (Stevens Institute of Technology, USA), who works as an adviser for businesses applying modelling. This paper combines and expresses the opinions of modellers working in different domains and using different modelling techniques, and represents the essence of the contributions of industrial participants of our workshop series. They all talked of the need to recognise and accumulate patterns and abstractions, and of the importance of ensuring that definitions are easily understandable by users. H. Kilov gives a historical example of specifications of insurance business showing that a complex specification is not a quality specification. The author calls on modellers to think clearly, separate essentials from specifics, avoid complexity and eschew tacit assumptions.

Standards in Behaviour Modelling

After this general introduction of modelling, the volume presents an example of using notations from standard UML for modelling behaviour.

The paper by Martin Gogolla, Lars Hamann, Frank Hilken, and Matthias Sedlmeier presents the evolution of a UML- and OCL–based (Object Constraint Language) tool for modelling. The group has been developing the tool USE (UML-based Specification Environment) for about 15 years. While it started as a tool for structural modelling with OCL constraints, it now addresses behaviour too. The paper illustrates the use of the tool for analysis of a system described structurally with a class diagram, including class invariants; and behaviourally with operation pre- and post-conditions, operation implementations, and statecharts. The paper shows that even for relatively small models, the validation of the structural models by behavioural views is needed, and is a non-trivial task. The authors talk of the need to work further to investigate how such tools would be used in the context of business systems development.

In the next paper, Gefei Zhang and Matthias M. Hölzl state that one of the barriers to widespread adoption of UML behaviour modelling languages is in the complexity of the models. The authors propose metrics of complexity of the UML statechart models arising from different kinds of non-locality for the current behaviour of a model being spread over several model elements instead of being locally available.

To illustrate the application of UML-based approaches for modelling embedded systems, we have chosen the paper by Karolina Zurowska and Juergen Dingel. They note that the execution semantics of UML behavioural models is not uniquely defined by the standard, and various different semantics have been proposed. They explore the possibility of a customisable execution engine that can be adapted to variations of the execution semantics, giving greater flexibility for developers to align their tooling to the task at hand.

New Ways of Behaviour Modelling: Events in Modelling

The next six papers of this volume present fresh ideas and concepts in modelling behaviour itself and for integrating it with classical modelling and programming mechanisms. We hope that these papers will inspire the modelling community to have a closer look at the potential of behaviour modelling and use them as a glue between the requirements descriptions and system implementations.

The contribution made by David Harel and Shani Nitzan exploits the application of the approach of Behaviour Programming that aligns the software development with descriptions of scenarios. The approach is applied to the domain of programs with animation. Behaviour Programming was proposed by the group led by David Harel. The basis of the approach is the language of Live Sequence Charts (LCS). LSCs extend Message Sequence Charts (MSC) with modalities in order to support the specification of liveness and safety properties and forbidden behaviours. The presented paper uses the ideas of Behaviour Programming in Java for the development of hybrid systems that combine discrete behaviour and continuous animation. The proposed approach integrates the local rules between various objects with the behavioural programming principles. As in hybrid automata, the states are governed by differential equations, enabling continuous behaviours between discrete state changes. The approach has a lot of application potential in modelling and simulation of interactive behaviour as its combines the analogous behaviour with easy changeable rule abstractions to switch such behaviour.

The paper by Jesper Jepsen and Ekkart Kindler proposes the Event Coordination Notation (ECNO) for modelling the desired behaviour of a software system on top of any object-oriented software. The authors emphasise the fact that the mechanism of method invocation built in the foundation of the modelbased development is quite different from the way of the observed object communication captured by behaviour models. Therefore, the authors build their own notation, ECNO, that allows one to model the behaviour of a system on top of structural models such as class diagrams. ECNO is based on the basic coordination mechanisms proposed by Hoare and Milner for defining interactions involving larger parts of a system. One of the main features of ECNO is that it allows modelling behaviour on top of existing object-oriented systems, and this way integrates with classical programming.

New Ways of Behaviour Modelling: Protocol Modelling

One of the core concepts used in the two papers above is that events are made a first class modelling concept. Once defined, events can be used to coordinate the behaviour of different components of the system. This idea is also a cornerstone of the Protocol Modelling approach proposed by Ashley McNeile, which represents behaviour using event-driven machines, called “protocol machines”, composed in the manner of Hoare’s CSP. The interest in Protocol Modelling at the workshops is evidenced by the fact that 14 contributions to the workshops made use of Protocol Modelling ideas; and

the editors have chosen three papers that illustrate different applications of these ideas. In order to make this book self-contained, we reprint the paper by Ashley McNeile and Nick Simons that introduces Protocol Modelling.

The paper by Marco Konersmann and Michael Goedicke highlights the features of the domain of modern information systems that usually do not use behaviour models and do not enjoy the benefits of reasoning about the behaviour and the ability to model small, interacting behavioural components. The authors predict, however, the need of models for future information systems and propose a framework, which imposes a low barrier for integration of models. The paper presents a new way of programming with protocol models integrated with source code. The protocol models replace the parts that are not implemented yet. The protocol models are executable and can be used for system debugging and testing at earlier design stages. A combination of models and the implemented code in one framework may give rise to a flexible system development approach.

The paper by Serguei Roubtsov and Ella Roubtsova investigates a particular way of system modularisation separating the most changeable parts of systems. The authors call them decision modules, and show how decision modules are expressed in requirements and behaviour models. Decision modules are formalised as protocol machines built using the event-based abstraction and possessing the unidirectional dependency with protocol machines. The paper presents an analysis of different Java implementation techniques (object composition, reflection, the publisher-subscriber design pattern, interceptors, and aspects) aimed at establishing the possibility of implementing decision modules having an event-based abstraction level and unidirectional dependency with other modules. The paper also discusses the functionality of a generic library that was developed by the authors for adopting the new style of modularisation of locally changeable implementations with separated decision modules.

In the paper contributed by the research group led by Jörg Kienzle (Wisam Al Abed, Matthias Schöttle, Abir Ayed, Jörg Kienzle), this compositional approach is used for aspect-oriented concern separation, and built into their tool Concern-Oriented REuse (CORE). CORE contains many other views traditionally used for the UML-based models. Classes statically define what functionality they offer. The message views show how instances of these classes interact with each other and with objects of other concerns to achieve this functionality. The state view complements the message views by using protocol machines to specify the order in which an object's operations should be called. CSP parallel composition is used to synchronise operation invocations. As the main purpose of CORE is model validation and documentation, CORE can combine different state views and generate new logically related protocol views useful for validation and documentation. The team developing CORE continues to experiment with the incorporation of Protocol Modelling ideas into model driven software development.

Conclusion

The contributions of this volume show that behaviour models can play a significant part in software development. These models can be on a high level of abstraction and very close to domain modelling and requirements modelling; still, they can be used for automatically executing them, for discussing and reasoning with them, and for verifying the correctness of the developed system. With increased tool support, this could lead to the development of implementation platforms which also make use of behaviour models. However, there is still some way to go to exploit the full potential of behaviour modelling — and we need to keep challenging existing notations and concepts, when they do not adequately serve the needs and purposes, so that eventually we will have distilled those concepts and notations that best serve their purpose.

Papers were contributed to the workshops by both European and American researchers, so the key terms: behaviour (behavior) and modelling (modeling) will appear in this volume both in British and American spelling.

The editors would like to thank:

- All the participants of the six international workshops for their contributions and sharing ideas during the workshops
- The invited speakers, Prof. Dr. Gregor Engels (University of Padderborn, Germany) and Michael Poulin (Head of Enterprise Architecture at Clingstone, Ltd. Bromley, UK)
- The authors of this volume
- The reviewers of the Reviewing Committee for their quality reviews and strong interest in the topic

In particular, we wish to thank the publisher, Springer, for making this publication proceedings possible.

June 2015

Ella Roubtsova
Ashley McNeile
Ekkart Kindler
Christian Gerth

References

1. BM-MDA 2009: Proceedings of the First International Workshop on Behaviour Modelling in Model-Driven Architecture. ACM, New York (2009). ISBN 978-1-60558-503-1
2. BM-FA 2010: Proceedings of the Second International Workshop on Behaviour Modelling: Foundation and Applications. ACM, New York (2010). ISBN 978-1-60558-961-9
3. BM-FA 2011: Proceedings of the Third Workshop on Behavioural Modelling. ACM, New York (2011). ISBN 978-1-4503-0617-1
4. BM-FA 2012: Proceedings of the Fourth Workshop on Behaviour Modelling - Foundations and Applications. ACM, New York (2012). ISBN 978-1-4503-1187-8

5. BMFA 2013: Proceedings of the 5th ACM SIGCHI Annual International Workshop on Behaviour Modelling - Foundations and Applications ACM, New York (2013). ISBN 978-1-4503-1989-8
6. BM-FA 2014: Proceedings of the 2014 Workshop on Behaviour Modelling-Foundations and Applications. ACM, New York (2014). ISBN 978-1-4503-2791-6

http://www.springer.com/978-3-319-21911-0

Behavior Modeling -- Foundations and Applications
International Workshops, BM-FA 2009-2014, Revised
Selected Papers
Roubtsova, E.; McNeile, A.; Kindler, E.; Gerth, C. (Eds.)
2015, XIV, 279 p. 101 illus., Softcover
ISBN: 978-3-319-21911-0