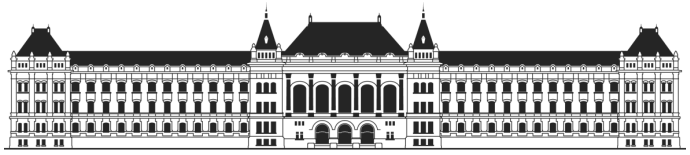

MŰEGYETEM 1782
Budapest University of Technology and Economics
Department of Measurement and Information Systems

Back-annotation of Execution Sequences by Advanced Search and Traceability Techniques

PhD thesis booklet

Ábel Hegedüs
MSc in technical informatics

Advisor:
Dr. Dániel Varró, DSc
associate professor

Budapest, 2014

1 Preliminaries

The advance of information technology gives rise to software systems with an ever increasing level of complexity. The development of such complex systems requires a coordinated effort of a lot of people from many areas in projects often running for several years. Well-established project management and software development methodologies are often applied in these development processes to ensure that each development step (e.g. requirement specification, system design, implementation, integration, testing, documentation, maintenance) is performed in the necessary detail and with the required inputs and outputs for other steps. However, postponing the verification of the system to later phases of the development can be very expensive because the cost (both in time and resources) of correcting any errors made in early phases is huge as it can affect subsequent steps. In order to find design errors in such systems, the analysis of their functional and non-functional requirements are needed from early on in the design phases. While functional requirements specify the behavior of the system and its components, non-functional requirements describe criteria about the quality of the system, such as security, safety, scalability, maintainability or usability.

As models become ubiquitous as the main artifacts in the development of such systems [BG05], the analysis of the system is also performed using different models. These models may represent the software or hardware architecture (e.g. UML [Obj11c], SysML [Obj10] or AADL [SAE09]), the external or internal behavior (e.g. UML statecharts [Obj11c], BPMN [Obj11a], BPEL [OAS07]), the reliability or performance of components (e.g. MARTE [Obj11b]). Furthermore, particular industries often make use of more specific modeling languages, such as AUTOSAR [AUT12] in the automotive or Integrated Modular Avionics (IMA) [Pri92] in the avionics domain.

1.1 Formal analysis of behavioral modeling languages

In *model-based analysis* (high-level) domain models are often transformed automatically into formal models to find design errors in the system early in the development process [Bon+01] (illustrated on Figure 1). These formal models are inputs of analyzer tools that perform verification of system requirements and provide results, for example a counter-example that specifies an execution trace leading to an unwanted state.

As system engineers frequently lack the skill and expertise in constructing and analyzing formal models, model-based analysis can effectively hide the theoretical details of mathematics and allow the engineers to use their well-known high-level models to specify the system.

This automation supports the development of *hidden (or invisible) formal methods*, where the designer of the domain model can use the analysis methods without knowing them in details [TSR03]. However, the designer can only improve on the system model through design optimization if the formal analysis results are translated to the design tool by back-annotation to guide the optimization.

Behavioral modeling languages, which capture the behavioral aspects of the system under design, play a special role, as the correctness of dynamic behavior is difficult to guarantee. In (discrete event) behavioral models, the system is evaluated along different *states* (snapshots). The evolution of the system is carried out in a discrete sequence of *steps*, which altogether form an *execution trace*.

Similarly to high-level engineering languages, there are a large number of analysis formalisms that are either designed for modeling similar systems or specifically tailored for a particular type of analysis (e.g. Petri nets od labeled transition systems for modeling distributed systems or process algebra for verifying communication protocols). The most common formal analysis techniques for checking properties that refer to the desired dynamic behavior of the system

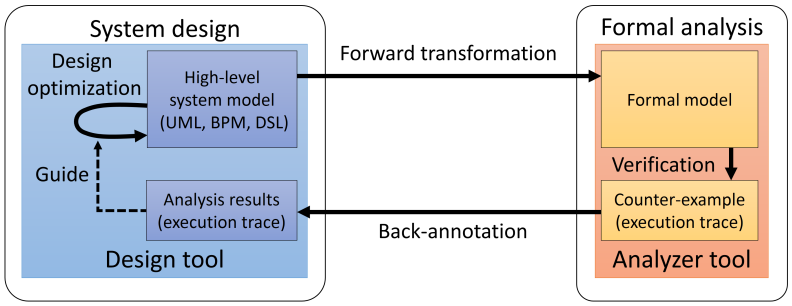

Figure 1: Model-based analysis [Bon+01]

are the following: (1) *static analysis* evaluates the model of the system only with symbolic execution [Aye+08], similarly (2) *theorem proving* uses an initial set of axioms and inference rules over the model for proving the property [HS96], (3) *model checking* starts from an initial state, exploring every reachable state of the system and checking that the property holds for each possible state and execution trace [Cla97], finally (4) *simulation* also starts from an initial state, but explores only some of the possible states and checks that the property holds over the traces traversed by the simulation [BCN10; Van08].

The optimization of the system model often necessitates a search-based approach for finding near-optimal configurations of various system quality attributes. *Design space exploration* (DSE) is a process to analyze several "functionally equivalent" implementation alternatives, which meets all *design constraints* in order to identify the most suitable design choice (*solution*) based on quality metrics such as cost or dependability. Design space exploration often appears as a challenging problem in application areas, such as dependable embedded systems [Moh+02; RLF08; Mee+11] and IT system management, where model-driven engineering (MDE) techniques are already popular. In addition, systems with modular software and hardware architectures (like AUTOSAR [AUT12]) introduced complex structural constraints and the dynamic creation and deletion of elements. Furthermore, DSE can be performed either during the design process to find optimal designs (see Figure 1) or during runtime to help dynamic reconfigurations.

1.2 Challenges in model-based analysis.

Automated model transformations are widely used for creating analysis models from design models for executing validation, verification, qualitative or quantitative analysis. Although numerous approaches were defined for analyzable model generation, they often lack solutions for processing analysis results automatically and thus fail to present them to the analyst in an intuitive, design model level representation. This reverse transformation problem, called *back-annotation* is non-trivial and appears as an additional challenge in most analysis techniques.

Storing the information regarding the correspondence of the design and analysis models is strongly related to back-annotation. *Traceability* is a common requirement in software development and specifically in model transformations. Several aspects of model-based design necessitate traceability information for either automatic execution or aiding user interaction. These aspects include multiple-phased transformations, requirement definition and back-annotation.

Back-annotation of execution traces. The **forward transformation** from domain model to formal model (see Figure 1) is usually performed by taking an input model describing a given system and deriving a precise formal description of the same system using the analysis language. Since the dynamic semantics of the domain and formal models are often considerably different,

the transformation needs to bridge a *semantic gap* between the domain model and the derived model.

Strictly from an analysis point of view, the aim of hidden formal analysis is to determine that the model meets the requirements specified for the system. From the engineering perspective, finding an execution trace that either proves or violates the property is essential in order to provide information that not only points out that the model is correct or not (yes/no answer) but also captures the particular sequence of steps that leads to this conclusion. In this sense, the results of both model checking and simulation are similar as they provide an execution trace that either violates the given property (**counter-example**) or conforms to it (e.g. a successfully completed simulation execution).

An execution trace provided by an analysis tool is often very complex and difficult to understand even with expert knowledge in the analysis language and the analyzed system. Therefore, it is important to support the designer of the system in interpreting the results of analysis in the context of the domain model that the designer is familiar with. **Back-annotation** (also known as back-propagation) is the backward translation of analysis results from the formal model to the domain model. It has been applied on feature models [CW07], UML object models [SAB09], Message Sequence Charts [Kov+09], security and performance models [Buc+05], web services [Fos+07] and embedded systems [CCS11] among others.

The back-annotation of execution traces from formal models to domain models involves capturing behavior by deriving an execution trace of the engineering domain model based on an execution trace of the formal model (see Figure 1). Although there is additional information available, such as the definition of the forward model transformation or the correspondence relations between the elements of the two models (i.e. which formal model element represents a given part of the domain model), the back-annotation problem still poses a number of challenges [16]. These challenges are mainly due to the semantic gap between the domain and formal models. Furthermore, it often turns out that the "wider" the semantic gap is, the harder it is to overcome the challenges in a particular back-annotation.

2 Research questions and challenges

My research has been motivated by the practical challenge to provide automated back-annotation of verification results from model checking to high-level engineering models.

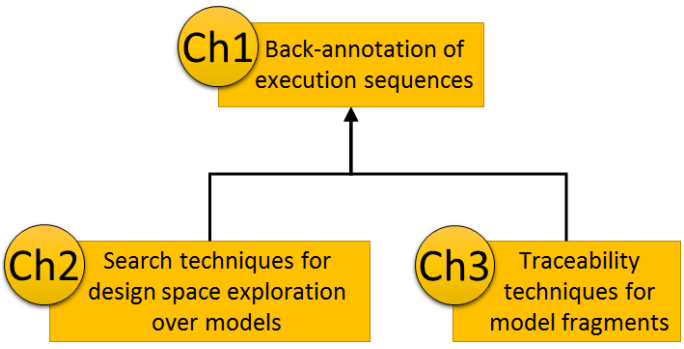

Figure 2: Research challenges and their relation

During the evaluation of existing techniques and the development of a *back-annotation approach for execution sequences* (Ch1) that can solve this practical challenge, I have identified two additional, tightly related research areas which are shown in Figure 2. The application of *search-based techniques* (Ch2) is often required for back-annotation, when the exploration of design

alternatives represented by models is necessary. Additionally, novel *traceability techniques* (Ch3) are needed for managing the correspondences between domain and formal models (model fragments).

Challenge 1: How to specify methods for the back-annotation of execution traces for behavioral modeling languages?

The methods defined for the back-annotation of execution traces are differentiated based on the availability of dynamic semantics for the engineering domain language.

For languages where the dynamic semantics is not defined, back-annotation methods need to include the specification of the possible execution steps of the language and provide a detailed mapping between steps of the formal language and the specified domain language steps. Therefore such methods are highly specialized for the selected domain and formal language pair and their application to a new pair of languages is difficult. An important challenge is to define how a generic method can be tailored to a specific pair of languages.

For behavioral modeling languages with precise dynamic semantics, back-annotation methods can take advantage of the semantics for deriving an execution trace, but also must ensure that the derived trace represents the same behavior as the input formal trace. The available semantics mean that the back-annotation can evaluate the possible execution steps when processing a trace. An important challenge is to *efficiently search through the possible execution traces* of the engineering model and evaluate whether it represents the dynamic behavior in the formal trace.

Challenge 2: How to improve design space exploration with advanced search techniques?

Design space exploration aims to construct and evaluate model alternatives in situations where the number of such alternatives is large and a systematic and automated enumeration of (some) potential solutions is necessitated. For example, the allowed operations of a behavioral modeling language specifies a design space that includes all possible execution traces starting from an initial model, while the goal of the exploration may be to satisfy all well-formedness constraints of the language. Indeed, the number of alternatives (i.e. the size of the design space) can be infinite with a high number of solutions (i.e. the solution space is dense), therefore it is preferable to guide the exploration towards solutions in order to increase the efficiency of the approach.

Existing design space exploration approaches often apply model checking with exhaustive state space exploration, solve finite domain constraint satisfaction problems or rely on heuristics driven by objectives and fitness functions. However, the exploration of model alternatives necessitates the development of advanced search techniques that are able to take advantage of the availability of allowed model manipulation operations, model queries and hints that are provided by model-based analysis tools.

Challenge 3: How to enable advanced traceability techniques for the interconnection of fragmented models?

Back-annotation involves, even in its most simple case, at least two separate models which are related to each other, since the target, formal model is prepared by a forward transformation from the source, engineering model. In order to identify elements in the engineering model that were used to create a given element in the formal model, this relation is stored by inter-model traceability links that are queried by back-annotation methods. Moreover, a large number of models are often used during the development of complex model-driven applications, which may include model transformations and back-annotation.

For maintainability and scalability reasons system models are frequently persisted as an interconnected network of model fragments where each fragment stores a certain part of the entire system model. In other application scenarios, complete models are used which are complemented with external traceability models to explicitly persist traceability links between requirements models, design models, analysis models or source code, for instance. In both scenarios, these models are frequently manipulated by several development or verification tools in complex tool chains operated by different design teams. It is an important challenge to provide traceability techniques that support these scenarios.

3 New Research Results

3.1 Modeling, replaying and back-annotation of execution traces

Execution traces capture the dynamic behavior of a system in a series of states and state transitions which can be later processed in order to ensure the correctness of the system or to analyse its performance. In Model-Driven Engineering behavioral models are increasingly used for specifying the behavior of the system-under-design and novel methods are needed to handle their execution traces in a model-driven context. I formulated the following thesis contributions related to the fulfillment of Challenge 1:

Contribution 1 *I defined back-annotation methods between execution traces of behavioral modeling languages, with or without operational semantics, using a model-based representation for execution traces and a generic replaying framework. Moreover, I integrated these methods for business process development to provide hidden formal verification.*

1. ***Model-based representation and generic replaying of execution traces.*** *I defined a model-based representation for execution traces and a generic replay technique for this trace representation [9, 6], which uses domain-independent model transformation rules to execute the captured dynamic behavior of the target model.*

2. ***Back-annotation of execution traces for behavioral languages without operational semantics.*** *I specified a method for the back-annotation of execution traces of behavioral models [16, 23, 9, 1], which uses traceability correspondence between the models for identifying related elements and change-driven model transformations to derive execution steps of the engineering model.*

3. ***Back-annotation of execution traces for behavioral languages with operational semantics.*** *I defined a search-based approach for the back-annotation of execution traces when the domain language has operational semantics [1], which uses traceability between the (semantic) operations of the languages and guided design space exploration for evaluating possible execution traces of the domain language.*

4. ***Hidden formal methods for business process verification.*** *I elaborated an approach for integrating forward model transformation and back-annotation for the hidden formal verification of business processes and presenting the verification results obtained by model checkers over Petri nets or transition systems as process execution traces [9, 25, 24].*

The model-based methods for the verification of business processes are also part of the PhD thesis of László Gönczy. The back-annotation methods builds on the concept of change-driven model transformations [Ber+12b] developed in details in the PhD thesis of Gábor Bergmann [Ber13] and István Ráth [Rát10] and the structural constraint satisfaction problems with domain specific manipulation operations, which is the foundation of the design space exploration over models [HV11] developed in details in the PhD thesis of Ákos Horváth [Hor13].

3.2 Guided design space exploration over models

To better exploit domain-specific information, designers often provide additional input (*hints*) about the system (e.g. from earlier experience or by some analysis) that can reduce the design space or at least start exploration with the most promising paths [Moh+02]. The design process is often complemented with different design, analysis and verification tools, which can also provide (mathematically well-founded) hints about the model in the early stages of development. These hints may express additional system properties, which can be incorporated in the DSE process to assist the evaluation of alternate solutions. I formulated the following thesis contributions related to the fulfillment of Challenge 2:

Contribution 2 *I defined an approach for guided design space exploration where hints obtained by system analysis are interpreted during the exploration to continue along promising search paths (using selection criteria) and to avoid the traversal of unpromising designs (by cut-off criteria). Moreover, I elaborated an algorithm that evaluates selection and cut-off criteria during exploration over a dependency graph of the allowed operations.*

1. ***Selection and cut-off criteria for design space exploration*** *I identified cut-off and selection criteria that are used for guiding design space exploration [7,22], where a selection criteria provides the next operation to be executed and cut-off criteria decides whether a state can never lead to a solution.*

2. ***Criteria evaluation over dependency graph.*** *I defined a dependency graph that combines dependency relations between allowed operations (captured as graph transformation rules) with occurrence vectors that specify how many times a rule should be applied and specified an algorithm that evaluates selection and cut-off criteria by using this dependency graph as a hint [7,14,22].*

3. ***Guided design space exploration.*** *I elaborated an approach to use cut-off and selection criteria evaluation on hints to guide the design space exploration process [14,13,21] and evaluated the efficiency of different exploration strategies.*

4. ***Quick fix generation for domain-specific models.*** *I specified a method for generating fix operations for correcting violations of well-formedness constraints in domain-specific models using design space exploration [13], where the hints are provided by evaluating the violations and applicability of the allowed operations.*

5. ***State encoding techniques for design space exploration.*** *I defined the challenges of state encoding to identify states that were already traversed by the exploration and described encoding techniques which can significantly reduce the design space [2].*

The implementation of the DSE framework both in VIATRA2 and EMF-INCQUERY builds on incremental pattern matching techniques [Rát+08; Ber+10], which are part of the PhD work of Gábor Bergmann [Ber13]. The Petri Net based abstraction technique is a work of Szilvia Varró-Gyapay and Dániel Varró [VV06] and its initial adaptation to DSE was done by Ákos Horváth [HV11]. My contribution is the definition of the dependency graph that incorporates the results of the Petri Net based analysis.

3.3 Soft interconnections for robust traceability

The application scenarios involving model fragments and complex tool chains demonstrate various shortcomings of modelling and traceability technologies [Kol+13; Ros+10]. First, circular traceability relations using only regular references often lead to serialization errors. Furthermore, without truly intelligent multi-resource transaction management, local changes in a model fragment may introduce broken traceability links unless all dependent model fragments are manipulated together in working memory. Such broken links require tool-specific resolutions — with a worst case scenario of fixing the links manually by the designer using text editors (and not the modeling tool). Finally, all traceability links captured by associations are explicitly persisted every time even if traceability links could be derived from existing unique identifiers. I formulated the following thesis contributions related to the fulfillment of Challenge 3:

Contribution 3 *I defined the concept of soft interconnections that provide inter-model links based on derived references computed by incremental model query evaluation and specified an approach for creating robust traceability between fragmented models using soft interconnections.*

1. ***Query-driven soft interconnection of model elements.*** *I defined soft interconnections represented as derived features (i.e. computed model attributes and references) driven by incrementally evaluated model queries, which behave transparently as regular model references [11, 10].*
2. ***Robust traceability in fragmented models.*** *I proposed an approach for providing traceability links between fragmented models using soft interconnections [10, 3], where the links can provide graceful degradation in case of major modifications. I also elaborated on how advanced model queries, soft interconnections and live validation of structural constraints can be used to provide an integrated traceability framework for complex domain-specific models.*
3. ***Experimental evaluation of soft interconnections.*** *I applied the soft interconnection approach for EMF models using EMF-INCQUERY and evaluated its applicability and performance on multiple case studies [3].*

4 Application of Results

The practical relevance of the methods and techniques outlined in the current thesis are demonstrated in this section by listing the applications of the results.

Tools I: VIATRA2 Design Space Exploration framework

The results of Contribution 2 were used to extend the VIATRA2 Design Space Exploration (VIATRA2 DSE) framework, an add-on for the VIATRA2 framework. These extensions include novel exploration strategies which use guidance from system analysis to increase the efficiency of the traversal, improvements to the search tree handling and to the state space encoding. We are also in the process of adopting the VIATRA2 DSE framework (see Contribution 2) to EMF using EMF-INCQUERY (in collaboration with my colleague Ákos Horváth and MSc students András Szabolcs Nagy and Miklós Földényi).

The VIATRA2 DSE framework was also successfully applied to new application domains for generating quick fixes for domain-specific modeling languages. The framework is available from `http://incquery.net/publications/dse`.

Tools II: EMF-INCQUERY query-based derived features

The EMF-INCQUERY framework adapts the incremental query evaluation techniques of VIATRA2 to EMF, one of the most widely used modeling environments today. The incremental query evaluation serves as a basis for additional features such as automated validation, data binding, query based features and model transformation. The incremental graph pattern matcher of the EMF-INCQUERY framework is a major research contribution of Gábor Bergmann's PhD thesis and while the project is lead by István Ráth.

I have implemented the query-based features in EMF-INCQUERY which serve as a basis for soft interconnections in Contribution 3. More details on the documentation page `http://incquery.net/incquery/new/examples/query-driven-soft-links`.

Applications I: End-to-end verification of BPEL with hidden formal methods

The results of Contribution 1 has been implemented into an end-to-end verification tool for business processes defined using the BPEL standard. This verification tool was developed as part of the SENSORIA EU FP6 project together with my colleague László Gönczy. The tool is integrated into the Eclipse framework and allows business process designers to execute model checking on a BPEL process by selecting or specifying requirements to be verified. In the background, the tool invokes the SAL model checker tool and parses the result which represents a counter-example if the requirement is violated. The counter-example is automatically processed and the designer can replay it as an animated execution on the business process editor directly.

Both the transformation from BPEL to SAL and the back-annotation from counter-example to BPEL execution are implemented using the VIATRA2 framework. The verification tool is available from the department's VIATRA2 site at `http://viatra.inf.mit.bme.hu/publications/exectraces`.

Applications II: Model-driven allocation for Integrated Modular Avionics

The results of Contribution 3 are applied in a cooperative research project between Embraer (a large airframer) and our research group. The aim of the project is to develop an application that allows engineers to use component libraries to specify the hardware architectures then allocate the functions of a functional architecture model to the hardware components. Finally, an integrated architecture model is created that is ready for environmental simulation.

This complex application is built using multiple, complex domain-specific models that are interconnected. Furthermore, the allocation process requires advanced model queries, well-formedness validation of structural constraints and strict traceability between artifacts.

5 Publication list of Ábel Hegedüs

Number of publications: 26
Number of peer-reviewed publications: 21
Number of independent citations: 36

International, peer-reviewed journal papers

[1] **Á. Hegedüs**, Á. Horváth, and D. Varró. "Back-annotation of execution sequences of behavioral models". In: *Systems and Software* (2014). Under review (after major revision)

[2] **Á. Hegedüs**, Á. Horváth, and D. Varró. "A Model-driven Framework for Guided Design Space Exploration". In: *Automated Software Engineering* (2014). Submitted after minor revision

[3] **Á. Hegedüs**, Á. Horváth, I. Ráth, R. R. Starr, and D. Varró. "Query-driven soft traceability links for models". In: *Software and Systems Modelling* (2014). Under review (after major revision)

[4] Z. Ujhelyi, **Á. Hegedüs**, G. Bergmann, Á. Horváth, I. Ráth, and D. Varró. "EMF-IncQuery: An Integrated Development Environment for Live Model Queries". In: *Science of Computer Programming* (2014). DOI: 10.1016/j.scico.2014.01.004

[5] E. Jakumeit, S. Buchwald, D. Wagelaar, L. Dan, **Á. Hegedüs**, M. Herrmannsdörfer, T. Horn, E. Kalnina, C. Krause, K. Lano, M. Lepper, L. Rose, S. Wätzoldt, and S. Mazanek. "A Comparison of Transformation Tools Based on the Transformation Tool Contest". In: *Science of Computer Programming* (2013). DOI: 10.1016/j.scico.2013.10.009

[6] **Á. Hegedüs**, I. Ráth, and D. Varró. "Replaying Execution Trace Models for Dynamic Modeling Languages". In: *Periodica Polytechnica Electrical Engineering and Computer Science* 56 (2012). DOI: 10.3311/PPee.7078

[7] **Á. Hegedüs**, Á. Horváth, and D. Varró. "Towards Guided Trajectory Exploration of Graph Transformation Systems". In: *ECEASST* 40 (2011). Ed. by T. Margaria, J. Padberg, and G. Taentzer. URL: http://journal.ub.tu-berlin.de/index.php/eceasst/article/view/583

[8] **Á. Hegedüs**, Z. Ujhelyi, I. Ráth, and Á. Horváth. "Visualization of Traceability Models with Domain-specific Layouting". In: *ECEASST* 32 (2010). Ed. by T. Margaria, J. Padberg, and G. Taentzer. URL: http://journal.ub.tu-berlin.de/index.php/eceasst/article/view/510

Book chapter

[9] L. Gönczy, **Á. Hegedüs**, and D. Varró. "Methodologies for model-driven development and deployment: an overview". In: *Rigorous software engineering for service-oriented systems*. Ed. by M. Wirsing and M. Hölzl. Vol. 6582. Springer-Verlag, 2011, pp. 541–560. DOI: 10.1007/978-3-642-20401-2_26

International conferences

[10] **Á. Hegedüs**, Á. Horváth, I. Ráth, and D. Varró. "Query-driven soft interconnection of EMF models". In: *15th ACM/IEEE International Conference on Model Driven Engineering Languages & Systems*. Vol. 7590. LNCS. Acceptance rate: 23%. Innsbruck: Springer, 2012. DOI: 10.1007/978-3-642-33666-9_10

[11] I. Ráth, **Á. Hegedüs**, and D. Varró. "Derived Features for EMF by Integrating Advanced Model Queries". In: *8th European Conference on Modelling Foundations and Applications*. Vol. 7349. LNCS. Acceptance rate: 31%. Kgs. Lyngby: Springer, 2012, pp. 102–117. DOI: 10.1007/978-3-642-31491-9_10

[12] G. Bergmann, **Á. Hegedüs**, Á. Horváth, Z. Ujhelyi, I. Ráth, and D. Varró. "Integrating Efficient Model Queries in State-of-the-art EMF Tools". In: *50th International Conference on Objects, Models, Components, Patterns (TOOLS Europe)*. LNCS. Acceptance rate: 31%. Prague: Springer, 2012. DOI: 10.1007/978-3-642-30561-0_1

[13] **Á. Hegedüs**, Á. Horváth, I. Ráth, M. C. Branco, and D. Varró. "Quick fix generation for DSMLs". In: *IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC*. Acceptance rate: 33%. IEEE Computer Society, 2011. DOI: 10.1109/VLHCC.2011.6070373

[14] **Á. Hegedüs**, Á. Horváth, I. Ráth, and D. Varró. "A Model-driven Framework for Guided Design Space Exploration". In: *26th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. ACM Distinguished Paper Award, Acceptance rate: 15%. IEEE Computer Society, 2011, pp. 173–182. DOI: 10.1109/ASE.2011.6100051

[15] G. Bergmann, **Á. Hegedüs**, Á. Horváth, I. Ráth, Z. Ujhelyi, and D. Varró. "Implementing efficient model validation in EMF tools". In: *26th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE Computer Society, 2011, pp. 580–583. DOI: 10.1109/ASE.2011.6100130

[16] **Á. Hegedüs**, G. Bergmann, I. Ráth, and D. Varró. "Back-annotation of Simulation Traces with Change-Driven Model Transformations". In: *8th International Conference on Software Engineering and Formal Methods (SEFM)*. Acceptance rate: 22%. 2010. DOI: 10.1109/SEFM.2010.28

International workshops

[17] B. Izsó, **Á. Hegedüs**, G. Bergmann, Á. Horváth, and I. Ráth. "PN2SC Case Study: An EMF-IncQuery solution". In: *TTC 2013: Sixth Transformation Tool Contest, Post-Proceedings*. Ed. by P. Van Gorp, L. M. Rose, and C. Krause. Vol. 135. EPTCS. Budapest, Hungary: Open Publishing Association, 2013, pp. 106–114. DOI: 10.4204/EPTCS.135.14

[18] **Á. Hegedüs**, Z. Ujhelyi, and G. Bergmann. "Solving the TTC 2011 Reengineering Case with VIATRA2". In: *TTC 2011: Fifth Transformation Tool Contest, Post-Proceedings*. Ed. by P. V. Gorp, S. Mazanek, and L. Rose. Vol. 74. EPTCS. Zürich, Switzerland: Open Publishing Association, 2011, pp. 136–148. DOI: 10.4204/EPTCS.74.13

[19] **Á. Hegedüs**, Z. Ujhelyi, and G. Bergmann. "Saying Hello World with VIATRA2 - A Solution to the TTC 2011 Instructive Case". In: *TTC 2011: Fifth Transformation Tool Contest, Post-Proceedings*. Ed. by P. V. Gorp, S. Mazanek, and L. Rose. Vol. 74. EPTCS. Zürich,

Switzerland: Open Publishing Association, 2011, pp. 302–324. DOI: 10.4204/EPTCS.74.25

[20] **Á. Hegedüs**, Z. Ujhelyi, G. Bergmann, and Á. Horváth. "Ecore to Genmodel case study solution using the Viatra2 framework". In: *Transformation Tool Contest (TTC '10)*. Ed. by P. V. Gorp, S. Mazanek, and A. Rensink. Malaga, Spain, July 2010

National conferences

[21] **Á. Hegedüs**. "A Model-driven Framework for Guided Design Space Exploration". In: *Proceedings of the 19th PhD Minisymposium*. Budapest University of Technology, Economics, Department of Measurement, and Information Systems. Budapest, Jan. 2012

[22] **Á. Hegedüs**. "Criteria Evaluation-driven State Space Exploration of Graph Transformation Systems". In: *Proceedings of the 18th PhD Minisymposium*. Budapest University of Technology, Economics, Department of Measurement, and Information Systems. Budapest, 2011, pp. 42–45

[23] **Á. Hegedüs**. "BPEL Verification: The Back-annotation Problem". In: *Proceedings of the 17th PhD Minisymposium*. Budapest University of Technology, Economics, Department of Measurement, and Information Systems. 2010, pp. 30–31

[24] **Á. Hegedüs**. "BPEL2.0 munkafolyamatok formális verifikációja". In: *Tavaszi Szél Konferenciakiadvány*. 2009

Technical reports and online content

[25] **Á. Hegedüs**, I. Ráth, and D. Varró. *From BPEL to SAL and Back: a Tool Demo on Back-Annotation with VIATRA2*. Tech. rep. SEFM'2010 "Posters and Tool Demo Session" Track. Consiglio Nazionale delle Ricerche (CNR), 2010. URL: http://puma.isti.cnr.it/linkdoc.php?idauth=1&idcol=1&icode=2010-ED-003&authority=cnr.isti&collection=cnr.isti&lanver=en

[26] **Á. Hegedüs**. *A model transformation-based approach for the Dependability analysis of UML-based system designs with maintenance*. Tech. rep. rcl090601. University of Florence, Dip. Sistemi Informatica, RCL group, June 2009. URL: http://dcl.isti.cnr.it/Documentation/Papers/entry-techRep-rcl090601-229.html

Acknowledgements

My research work was partially supported by the Hungarian CERTIMOT (ERC_HU-09-1-2010-0003) project, the European Union SENSORIA (IST-3-016004), SecureChange (ICT-FET231101), CECRIS (324334) and MONDO (ICT-611125) research projects, the TÁMOP (4.2.2.B-10/1–2010-0009, 4.2.2.C-11/1/KONV-2012-0001) grants and a collaborative project with Embraer. My publishing was partially supported by the Schnell László Foundation.

References

[AUT12] AUTOSAR Consortium. *The AUTOSAR Standard*. http://www.autosar.org/. 2012.

[Aye+08] N. Ayewah, D. Hovemeyer, J. Morgenthaler, J. Penix, and W. Pugh. "Using Static Analysis to Find Bugs". In: *Software, IEEE* 25.5 (2008), pp. 22–29. DOI: 10.1109/MS.2008.130.

[BCN10] J. Banks, J. S. Carson, and B. L. Nelson. *Discrete-event system simulation*. 2010.

[Ber+10] G. Bergmann, Á. Horváth, I. Ráth, and D. Varró. "Incremental Evaluation of Model Queries over EMF Models". In: *Proc. of MODELS'10, ACM/IEEE 13th International Conference On Model Driven Engineering Languages And Systems*. 2010.

[Ber+12b] G. Bergmann, I. Ráth, G. Varró, and D. Varró. "Change-driven Model Transformations: Change (in) the Rule to Rule the Change". In: *Journal of Software and Systems Modeling* (2012).

[Ber13] G. Bergmann. "Incremental Model Queries in Model-Driven Design". PhD dissertation. Budapest University of Technology and Economics, 2013.

[BG05] S. Beydeda and V. Gruhn. *Model-Driven Software Development*. Springer-Verlag New York, Inc., 2005.

[Bon+01] A. Bondavalli, M. Dal Cin, D. Latella, I. Majzik, A. Pataricza, and G. Savoia. "Dependability analysis in the early phases of UML-based system design". In: *Computer Systems Science and Engineering* 16.5 (2001), 265–275.

[Buc+05] M. Buchholtz, S. Gilmore, V. Haenel, and C. Montangero. "End-to-end integrated security and performance analysis on the DEGAS choreographer platform". In: *Proceedings of the 2005 international conference on Formal Methods*. LNCS. 2005, pp. 286–301. DOI: 10.1007/11526841_20.

[CCS11] F. Ciccozzi, A. Cicchetti, and M. Sjodin. "Towards a Round-Trip Support for Model-Driven Engineering of Embedded Systems". In: *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on*. 2011, pp. 200–208. DOI: 10.1109/SEAA.2011.39.

[Cla97] E. Clarke. "Model checking". In: *Foundations of Software Technology and Theoretical Computer Science*. Vol. 1346. Lecture Notes in Computer Science. 10.1007/BFb0058022. 1997, pp. 54–56.

[CW07] K. Czarnecki and A. Wasowski. "Feature Diagrams and Logics: There and Back Again". In: *Proceedings of the 11th International Software Product Line Conference*. 2007, pp. 23–34. DOI: 10.1109/SPLC.2007.19.

[Fos+07] H. Foster, S. Uchitel, J. Magee, and J. Kramer. "WS-Engineer: A Model-Based Approach to Engineering Web Service Compositions and Choreography". In: *Test and Analysis of Web Services*. 2007, pp. 87–119.

[Hor13] Á. Horváth. "Search-Based Techniques in Model-Driven Engineering". PhD dissertation. Budapest University of Technology and Economics, 2013.

[HS96] K. Havelund and N. Shankar. "Experiments in theorem proving and model checking for protocol verification". In: *FME'96: Industrial Benefit and Advances in Formal Methods*. Vol. 1051. Lecture Notes in Computer Science. 1996, pp. 662–681. DOI: 10.1007/3-540-60973-3_113.

[HV11] Á. Horváth and D. Varró. "Dynamic constraint satisfaction problems over models". In: *Software and Systems Modeling* (2011). 10.1007/s10270-010-0185-5.

[Kol+13] D. S. Kolovos, L. M. Rose, N. Matragkas, R. F. Paige, E. Guerra, J. S. Cuadrado, J. De Lara, I. Ráth, D. Varró, M. Tisi, and J. Cabot. "A Research Roadmap Towards Achieving Scalability in Model Driven Engineering". In: *Proceedings of the Workshop on Scalability in Model Driven Engineering*. BigMDE '13. 2013, 2:1–2:10. DOI: 10.1145/2487766.2487768.

[Kov+09] T. Kovše, B. Vlaovič, A. Vreže, and Z. Brezočnik. "Spin Trail to Message Sequence Chart Conversion Tool". In: *Telecommunications, 2009. ConTEL 2009. 10th International Conference on*. 2009, pp. 125–129.

[Mee+11] I. Meedeniya, B. Buhnova, A. Aleti, and L. Grunske. "Reliability-driven deployment optimization for embedded systems". In: *Journal of Systems and Software* 84.5 (2011), pp. 835–846.

[Moh+02] S. Mohanty, V. K. Prasanna, S. Neema, and J. Davis. "Rapid design space exploration of heterogeneous embedded systems using symbolic search and multi-granular simulation". In: *SIGPLAN Not.* 37 (7 2002), pp. 18–27.

[OAS07] OASIS. *Web Services Business Process Execution Language Version 2.0 (OASIS Standard)*. "http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html". 2007.

[Obj10] Object Management Group. *OMG System Modeling Language (SysML)*. http://www.omg.org/spec/SysML/index.htm. 2010.

[Obj11a] Object Management Group. *Business Process Model And Notation (BPMN)*. http://www.omg.org/spec/BPMN/index.htm. 2011.

[Obj11b] Object Management Group. *UML Profile For MARTE: Modeling And Analysis Of Real-Time Embedded Systems*. http://www.omg.org/spec/MARTE/index.htm. 2011.

[Obj11c] Object Management Group. *Unified Modeling Language (UML)*. http://www.omg.org/spec/UML/index.htm. 2011.

[Pri92] P. Prisaznuk. "Integrated modular avionics". In: *Aerospace and Electronics Conference, 1992. NAECON 1992., Proceedings of the IEEE 1992 National*. 1992, 39–45 vol.1. DOI: 10.1109/NAECON.1992.220669.

[Rát+08] I. Ráth, G. Bergmann, A. Ökrös, and D. Varró. "Live Model Transformations Driven by Incremental Pattern Matching". In: *Theory and Practice of Model Transformations*. Vol. 5063/2008. LNCS. 2008, pp. 107–121. DOI: 10.1007/978-3-540-69927-9_8.

[Rát10] I. Ráth. "Event-driven Model Transformations in Domain-Specific Modeling Languages". PhD thesis. Budapest University of Technology, Economics, Department of Measurement, and Information Systems, 2010.

[RLF08] B. Ristau, T. Limberg, and G. Fettweis. "A mapping framework for guided design space exploration of heterogeneous MP-SoCs". In: *Design, Automation and Test in Europe, 2008. DATE '08*. 2008, pp. 780–783. DOI: 10.1109/DATE.2008.4484910.

[Ros+10] L. Rose, D. Kolovos, N. Drivalos, J. Williams, R. Paige, F. Polack, and K. Fernandes. "Concordance: A Framework for Managing Model Integrity". In: *Modelling Foundations and Applications*. Vol. 6138. LNCS. 10.1007/978-3-642-13595-8_20. 2010, pp. 245–260.

[SAB09] S. M. A. Shah, K. Anastasakis, and B. Bordbar. “From UML to Alloy and back again”. In: *MoDeVVa ’09: Proceedings of the 6th International Workshop on Model-Driven Engineering, Verification and Validation*. 2009, pp. 1–10.

[SAE09] SAE International. *Architecture Analysis & Design Language (AADL)*. `http://standards.sae.org/as5506a/`. 2009.

[TSR03] A. Tiwari, N. Shankar, and J. Rushby. “Invisible formal methods for embedded control systems”. In: *Proceedings of the IEEE* 91.1 (2003). Ed. by S. Sastry, J. Sztipanovits, R. Bajcsy, and H. Gill, pp. 29–39.

[Van08] H. Vangheluwe. “Foundations of Modelling and Simulation of Complex Systems”. In: *ECEASST* 10 (2008).

[VV06] S. Varró-Gyapay and D. Varró. “Optimization in Graph Transformation Systems Using Petri Net Based Techniques”. In: *ECEASST* 2 (2006). Ed. by P. Baldan, H. Ehrig, J. Padberg, and G. Rozenberg. Proc. of PNGT ’06.