MegaM@Rt^2 Project: Mega-Modelling at Runtime - Intermediate Results and Research Challenges

Andrey Sadovykh[1,3(✉)], Dragos Truscan[2], Wasif Afzal[4], Hugo Bruneliere[5], Adnan Ashraf[2], Abel Gómez[6], Alexandra Espinosa[4], Gunnar Widforss[4], Pierluigi Pierini[7], Elizabeta Fourneret[8], and Alessandra Bagnato[1]

[1] Research and Development Department, Softeam, Paris, France
{andrey.sadovykh, alessandra.bagnato}@softeam.fr
[2] Åbo Akademi University, 20520 Turku, Finland
{dragos.truscan, adnan.ashraf}@abo.fi
[3] Innopolis University, Innopolis, Russia
a.sadovykh@innopolis.ru
[4] Mälardalen University, Västerås, Sweden
{wasif.afzal, alexandra.espinosa.hortelano, gunnar.widforss}@mdh.se
[5] IMT Atlantique, LS2N (CNRS) & ARMINES, Nantes, France
hugo.bruneliere@imt-atlantique.fr
[6] Internet Interdisciplinary Institute, Universitat Oberta de Catalunya, Barcelona, Spain
agomezlla@uoc.edu
[7] Intecs Solutions S.p.A., Rome, Italy
pierluigi.pierini@intecs.it
[8] Smartesting, Paris, France
elizabeta.fourneret@smartesting.com

Abstract. MegaM@Rt^2 Project is a major European effort towards the model-driven engineering of complex Cyber-Physical systems combined with runtime analysis. Both areas are dealt within the same methodology to enjoy the mutual benefits through sharing and tracking various engineering artifacts. The project involves 27 partners that contribute with diverse research and industrial practices addressing real-life case study challenges stemming from 9 application domains. These partners jointly progress towards a common framework to support those application domains with model-driven engineering, verification, and runtime analysis methods. In this paper, we present the motivation for the project, the current approach and the intermediate results in terms of tools, research work and practical evaluation on use cases from the project. We also discuss outstanding challenges and proposed approaches to address them.

Keywords: Cyber-Physical systems · Model-Driven Engineering · Runtime Analysis · Tools · Mega-Modelling · Traceability · ECSEL

M. Mazzara et al. (Eds.): TOOLS 2019, LNCS 11771, pp. 393–405, 2019.
https://doi.org/10.1007/978-3-030-29852-4_33

1 Introduction

Electronic systems are becoming more and more complex and software intensive. This situation calls for modern software and systems engineering practices in order to keep high productivity and quality levels. In the last decade, the ecosystem around Model-Driven Engineering (MDE) has flourished, providing developers with a plethora of tools. However, these tools need to be further developed to scale up for real-world industrial applications. They also need to be enhanced in order to provide advantages at runtime as well. This represents a real opportunity for achieving a complete continuous systems engineering lifecycle, thus connecting together the design and runtime phases [1, 2].

The MegaM@Rt^2 project's main goal is to create a framework incorporating methods and tools for the continuous development and runtime support of complex software-intensive systems. Our current architecture vision and development over the MegaM@Rt^2 framework integrate three main complementary big capabilities: systems design engineering, runtime analysis, and global model & traceability management. The project is organized around the research work and related technical developments concerning the tool sets supporting those capabilities.

The research topics include holistic Systems Engineering covering design, verification and validation; Runtime Analysis dealing with monitoring, online testing and verification as well as models@runtime techniques; and so-called Mega-Modelling, i.e. large-scale model and traceability management. The framework is under evaluation by 9 industrial case studies ranging from transportation - avionics, railway, automotive, traffic monitoring; and telecommunications - short range communications, base transceiver stations; to logistics - indoor positioning, smart warehouses domains. Among the partners providing use cases in the project, we can cite Thales, Volvo Construction Equipment, Bombardier Transportation and Nokia. These organizations have different product management and engineering practices, as well as regulatory and legal constraints. This results in a large and complex catalog of requirements to be realized by the architecture building blocks at different levels of abstraction. Thus, the development of the MegaM@Rt^2 framework is based on a feature-intensive architecture and on a related implementation roadmap that is kept up-to-date. A comprehensive set of the project information, as well as the published deliverables, are all publicly available from the project web site [3].

In this paper, we present the main project research and technological results after two years, and outline the outstanding challenges and further work. To this intent, the rest of the paper is structured as follows. Section 2 briefly describes the MegaM@Rt^2 overall approach. Then Sect. 3 focuses on the three complementary tools sets that are designed and developed in the project to support this MegaM@Rt^2 approach in practice. We notably insist on the main related research achievements we obtained so far, as well as on still open research and technical challenges. Finally, Sect. 4 concludes by summarizing the main results from this first phase of MegaM@Rt^2 and by opening on some future work to come during the second phase of the project.

2 Description of the Overall Approach

As stated in the Electronic Components and Systems for European Leadership program's Multi-Annual Strategic Plan [4], design methods and related technologies should fully support the constant technology push and corresponding new user/society demands of products/services based on more and more complex Electronic Components and Systems (ECS). This is particularly true in the context of the involved software components relying on hardware configurations and their interactions e.g. with their underlying environment, being very often numerous, complex, heterogeneous and strongly interrelated. In the past, Model-Based Engineering principles and techniques have already shown promising capabilities that have been experimented in such context. However, they have generally failed in terms of (1) scalability to support real-world scenarios implied by the full deployment and use of complex ECS and (2) efficient traceability, integration and communication between two fundamental system levels which are design time and runtime, notably as far as non-functional properties and their verification & validation aspects are concerned.

As a consequence, the overall idea of MegaM@Rt^2 is to scale up the use of model-based techniques by offering scalable methods and related tools interacting between both design time and runtime, as well as to validate the designed and developed approach in concrete industrial cases involving complex ECS. To this intent, MegaM@Rt^2 proposes an overall model-based approach combining existing techniques to be enhanced when relevant, and novel ones to be developed when needed. A fundamental challenge notably resides in providing efficient traceability support between the two levels i.e. from design models to runtime ones and back. In parallel to these, modern large-scale industrial software engineering processes require thorough configuration and model governance to provide the promised productivity gains. Thus, a scalable mega-modelling approach is being designed and will be deployed to manage all the involved artifacts e.g. the many different models, corresponding workflows, configurations, etc. and to better tackle their large diversity in terms of nature, number, size, complexity, etc.

To cover all these topics and deal with the complete value chain, MegaM@Rt^2 brings together prominent tool developers and vendors and research organisations with state-of-the-art methods and tools that are validated in highly relevant European industry case studies. The end users from the space, naval, railway, smart grid, smart warehouse and telecom industry domains are driving the project by providing real-world requirements and case studies as well as by validating and endorsing the MegaM@Rt^2 results.

Figure 1 provides an overview of the MegaM@Rt^2 global approach and emphasizes its key principles and concepts. Industries apply a set of current engineering practices based on SysML, AADL, EAST_ADL, but also Matlab/Simulink, and Method B, each one producing specific design models, requirement specifications and resulting software and hardware artefacts. MegaM@Rt^2 suggests to integrate those artefacts into a global system model providing a complete view of the Cyber-Physical System (CPS), and detailing the component, behaviour and desired quality properties of the system. These properties are then an object of exhaustive continuous testing and

monitoring in the runtime environment to detect deviations in real-time, thanks to the configuration of the target platform and the injection of probes in the software. The detected deviations plus all the traces information collected in the process are analyzed to detect the impacted components in the integrated view of system models. When possible, automatic repairing suggestions are provided to correct the issue and reconfigure or redeploy the system to start the next iteration of the continuous integration process. This approach was further developed in [5] where we defined the specific tool sets - their requirements and features as well as outlined integration means.

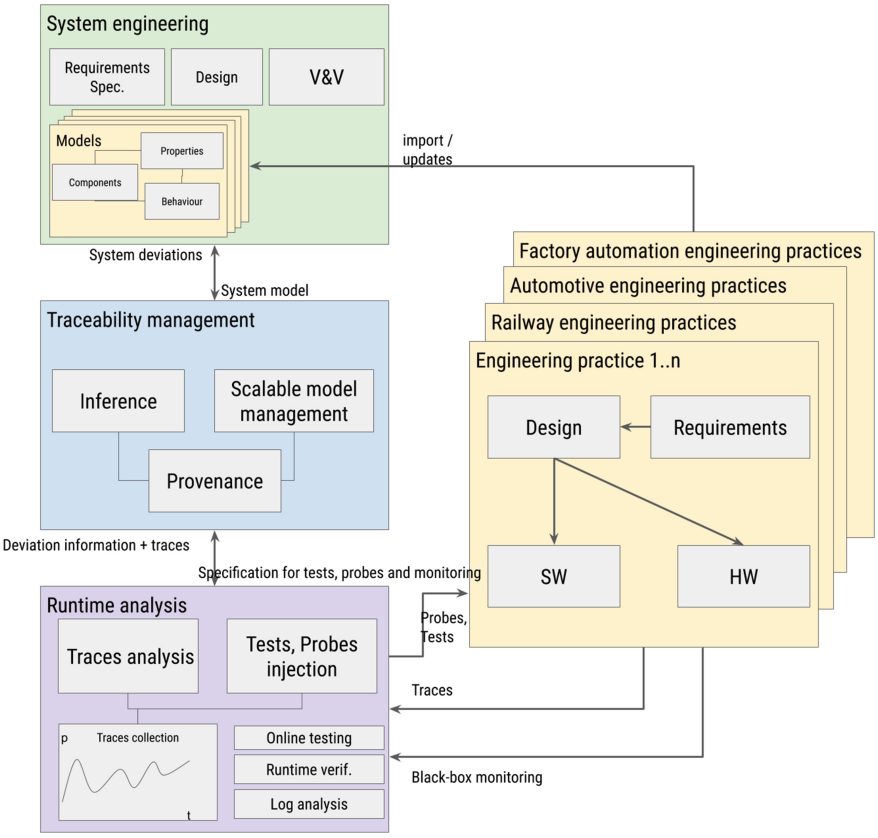

Fig. 1. Overall conceptual architecture of the MegaM@Rt project.

The methods and tools provided by MegaM@Rt[2] are evaluated and applied in several industrial case studies. Each individual case study defines a set of key performance indicators (KPIs) that are used to evaluate the improvement that the new technologies provide. The case study specific KPIs are aggregated into project level KPIs which provide a quantitative evaluation of the project goals.

The project has set challenging goals in terms of KPIs such as:

- Reduction of design time/design effort in the range of 10%–50% by design artefacts reuse.
- Reduction of validation effort in the range of 10%–30% by automated trace collection and analysis.
- Reduction 10%–50% in time/effort required for managing and handling all the involved models (e.g. time for model retrieval and access).
- Reduction 10%–50% in time/effort required for tracing and handling all the involved models at design and runtime levels (e.g. creation of and access to relations between system and traces models).

The above-mentioned KPIs are measured through out the project industrial case studies. At the current stage the first evaluation phase has finished. The next sections present the preliminary results.

3 Results of the First Evaluation Phase and Outstanding Research Challenges

At the time of writing this paper, the MegaM@Rt^2 project has entered its second half. In the following sections, we provide an overview of the current achievements of the project by focusing on research work and the corresponding results that we have already obtained.

3.1 The MegaM@Rt^2 System Engineering Tool Set

This tool set aims to support system design activities. It has been architected around three main topics: (i) requirements analysis & specification, (ii) system modeling and (iii) model verification & validation. The approach integrates up to 20 different open source tools, mainly Eclipse-based, such as Modelio, developed by the consortium research partners. These tools support a variety of current engineering practices based on standard modelling languages, profiles and extensions like: UML, SysML, MARTE, AADL, EAST_ADL, etc. The framework is designed to integrate additional "external" tools like Matlab/Simulink, AUTOSAR, Modelica and others, based on specific needs of the industrial partners.

Different techniques have been adopted to ensure the correctness of system models, either in terms of verification of languages syntactic paradigms (e.g. using SAT- and CP-solver technologies) and in terms of functional and non-functional validation of system artefacts with respect of given requirements (e.g. through model simulation, model testing, machine learning technique, etc.). For example, [6] proposes a framework to reason about the satisfiability of class models described using the Unified Modeling Language (UML). It allows to identify possible design flaws as early as possible in the software development cycle, by annotating UML Class Diagrams with Object Constraint Language (OCL) invariants. Then, the Constraint Logic Programming (CLP) paradigm allows to reason about UML Class Diagrams modeling foundations thanks to a translation to Formula.

Several other research areas have been investigating. For instance, the current trend on Internet-of-Things Systems of Systems (IoT-SoS) implies significant evolution of modeling, analysis and design approaches [7].

Separation of concerns is one of the fundamental principles allowing to build well-structured software and improving its maintainability/evolutivity. Executable models are good candidates to capture the behavior of a software-intensive system using separation of concerns approach. In [8], Domains Specific Languages (DSL) have been exploited to create executable models when business operations are tied to specific technological platforms. This method is applied both at design-time for creation of executable models with EMF and at run-time by monitoring operation calls from the deployed execution engine.

Another aspect investigated in [9, 10] is the availability of platform-independent SW models and HW synthesis tools able to automatically produce efficient implementations based on performance predictions of the system model and this on many different distributed and parallel computing resources.

Safety critical systems, e.g. as proposed by Bombardier Transportation and ClearSy in the project, require specific support for safety analysis, assessment and certification. The contract-based approach is adopted by some of the framework tools and is presented and discussed in [11]. It is based on finding static schedules relying on contracts and using this information in the verification process to reduce the number of invariant annotations needed. Moreover, contracts can be used to make compile-time scheduling decisions, improving runtime performance.

A complementary research area is related to the application of the Aspect-Oriented Methodologies focusing on the reduction of the modeling and verification effort by applying aspect-oriented principles in model construction [12]. The industrial partners have a preference for more classical and consolidated methodologies. However, such capabilities are still available for possible future applications in case needed.

In the general case, a main achievement is the ongoing contribution to the MARTE standard, as presented in [13] and responding to the Request for Information issued by the OMG for a new MARTE 2.0. Partners proposals have been collected in an initial survey, then an answer to the RFI has been prepared and sent back.

Finally, the last project period will focus on the exploitation, at design time, of the runtime trace collection and analysis capability, in order to address possible model refinements in the context of feedback loops. To this intent, the most promising approach is the one provided by PADRE tool on performance anti-pattern detection and model refactoring [14, 15]. Finally, as a part of an effort to automate system engineering, [16] provides a systematic mapping study on published tools and approaches that can be used for generating API documentation, or for assisting in the API documentation process. the paper presents an overview of what kind of tools have been developed, what kind of documentation they generate, and what sources the documentation approaches require.

3.2 The MegaM@Rt^2 Runtime (Trace) Analysis Tool Set

This tool set aims to define new methods and tools for creating and managing models at runtime verification and testing, including automated runtime testing and monitoring as

well as a model-based log collection and analysis infrastructure supported by tools such as PauWare or CertifyIt. This runtime tool set integrates 24 tools that further propose automated code generation, model execution as a part of a system, runtime verification and online testing, such as CompleteTest, JTL, PauWare, Smartesting tools, AIPHS, Comformiq Designer, Modelio, etc. These tools within the MegaM@Rt^2 approach integrate with the analysis tools. The main ongoing activity is related to establishing a smooth connection with the analysis tools, that will allow user-friendly and simple inclusion within the continuous development process, addressed in MegaM@Rt^2.

Several results have been published. In the context of testing and test generation several papers address test generation using UPPAAL model checker and its extensions. For instance, [17] outlines a method for testing energy consumption in embedded systems using energy-related mutants for EAST-ADL architectural models, which are converted to UPPAAL Timed Automata and used for test generation UPPAAL Statistical Model Checker (SMC). A complementary approach is presented in [18], where we show how architectural models described in the EAST-ADL architectural language can also be used for testing the energy consumption of embedded systems, after transforming them into networks of formal models called priced timed automata. A mutation testing approach for UPPAAL TA has been proposed in [19] to mutate UPPAAL-TA models and use them for generating tests used for evaluating security vulnerabilities of web services. Last but not least, in order to enable the analysis of failed traces and quick fault localization, [20] proposes an approach that converts concrete test sequences generated and executed by Uppaal Tron against the system under test into symbolic traces that can be imported in the Uppaal tool and visualized in the Uppaal simulator.

In the same context of testing, [21] presents an approach for testing of software intensive safety-critical products to validate the hardware-in-the-loop simulation of a safety-critical system, by executing test cases both in the control setting (lab) and on the real product (train). The process is intended to be used when certifying the simulation which is a necessary step in order to certify the complete system. In addition, in [22], the authors propose an extension of base-choice criterion used for testing software-based on its nominal choice of input parameters, which takes into account time as another parameter when generating and executing tests by defining the timed base-choice coverage criterion. In [23], the authors conducted a comparative study on the cost and effectiveness of tests that are manually written versus those that are automatically generated in the field of industrial control software, where strict requirements on both specification-based testing and code coverage typically are met with rigorous manual testing.

In order to explore the performance of deployed systems at runtime, [24] suggests a performance space exploration approach for inferring the worst-case user scenario in a given workload model. The goal of this work is to detect which configuration of the load model has the potential to create the highest resource utilization on the system under test with respect to a given resource so that performance tests can be run with that configuration. An exact and an approximate method are suggested and compared.

Finally, in [25] we propose a marker design and an algorithm to detect the markers under different ambient conditions, with a long range to be executed on embedded systems with low computational requirements. The proposed method reduces the

existing problems in the state-of-the-art related to the use of different environments and conditions such as different distances or different illumination.

3.3 The MegaM@Rt^2 Model and Traceability Management (MTM) Tool Set

The Model and Traceability Management (MTM) tool set aims at providing generic global model management and traceability capabilities, with a focus on the dedicated support for creating and using feedback loops between design-time and runtime models in the context of complex CPSs engineering. To this intent, the MTM tool set is composed of 5 different complementary tools supporting the Eclipse [26] and Modelio [27] technical modeling environments. These tools provide support for storing and handling large EMF models (NeoEMF) [28], building and handling views integrating different EMF models (EMF Views) [29], keeping consistency and traceability between different EMF models (JTL) [30], detecting and refactoring performance antipatterns (PADRE) [14] or organizing and managing Modelio-based models and their relationships (Modelio Constellation) [31]. In all cases, their main objective is notably to leverage the different kinds of models resulting from the System Engineering and Runtime Analysis tool sets, in order to handle and reuse these models altogether in a coherent way as part of the continuous CPS engineering approach promoted by MegaM@Rt^2. During the first phase of the MegaM@Rt^2 project, a significant research effort has been conducted by the involved partners in order to provide these fundamental capabilities via the various tools of the MTM tool set. We summarize significant related research achievements in what follows.

On one hand, we have worked on improving the general support for backward traceability and change propagation between different kinds of models thanks to the JTL tool [32]. We then used such a support in order to provide change propagation capabilities at architectural (design) model-level, and illustrated it in a software availability context [33]. We also used this same support in order to automate performance improvements via the detection of architectural antipatterns using PADRE and thanks to traceability with corresponding runtime data [15] (cf. also Sect. 3.1).

On the other hand, we have obtained interesting results in the model view area [34]. Notably, we have worked on supporting the creation and handling of scalable model views combining different large-scale models together (including design and runtime ones) via traceability links [35]. To this intent, we worked on providing the required infrastructure to store, handle and trace efficiently very large models. This has been implemented in practice by leveraging the EMF Views and NeoEMF tools from the MTM tool set. This was a required achievement in order to be able to implement runtime-to-design time feedback loops, which is one of the longer-term objectives of MegaM@Rt^2.

Interestingly, based on the two complementary efforts above-mentioned, we have then been able to apply our model view approach - EMF Views, in combination with our traceability capabilities in JTL, in order to provide a first concrete instantiation of the MegaM@Rt^2 runtime-to-design feedback loop in the context of a safety-critical system from our partner ClearSy [36]. In the second phase of the project, we plan to

work on more practical instantiations of such a feedback loop by relying on tools from the MTM tool set.

Nevertheless, there are still open challenges in these promising research areas. We have already been able to discuss that within the Modeling community when organizing and running the first edition of the International Workshop on Model-Driven Engineering for Design-Runtime Interaction in Complex Systems (MDE@DeRun 2018), co-located with STAF 2018 in Toulouse, France [37]. Notably, we identified challenges related to the particularities of design-runtime traceability: e.g. which semantics has to be given to the traceability information, in which contexts and how? We also identified questions related to the analysis of the traced runtime information: e.g. what kind of runtime data is actually needed, in which contexts and how to collect it properly? Finally, we identified issues related to the overall objectives of such a design-runtime traceability: e.g. which engineering purposes or activities do we intend to address or cover thanks to such feedback loops?

3.4 Case Study Evaluation

A total of nine industrial case studies are used in the project in order to evaluate the MegaM@Rt^2 framework in practice. To provide measurable evidence on the extent to which the framework fits and provides benefits to the industrial development process, each case study defined a set of Key Performance Indicators (KPIs) that have been measured at baseline (i.e. when the project started) and have been/will be measured again after each of the two development phases of the project (i.e. at month 24 and month 36 respectively). At the time of writing this paper, the first evaluation phase has recently finished (at month 24).

During Phase 1, the case study providers have evaluated different scenarios using the tools and technologies offered by the different tool sets previously presented in this Sect. 3. The evaluation in terms of scenarios has put the focus on the benefits that MegaM@Rt^2 is expected to bring: (1) They allow to better understand the aspects that the case study providers found most important for their industrial activities and (2) They structure and organize the tools' verification and validation, which are based on the requirements and the KPIs defined by MegaM@Rt^2. The case study providers made some changes in the choice of the best scenario to validate a tool/technology and, conversely, in the judgement of the best way for using a tool in a certain scenario; many problems were encountered but solved thanks to the collaboration that the tool providers fully offered.

The details on the case study results are provided in deliverable D5.5 as available from the project website [38]. It is important to mention that case studies measure differently the KPIs depending on their respective contexts and designed experiments. Nevertheless, it is noteworthy to point out that the case studies succeeded to demonstrate improvements significantly above targets, in particular, in:

- Time required for identification of design problems;
- Time/effort for requirements validation;
- Productivity improvements;
- Cost savings for development and maintenance of large complex systems.

The project has also already demonstrated values close to the targets for the following set of KPIs:

- Reduction of validation effort by automated trace collection and analysis;
- Reduction in time/effort required for tracing and handling all involved models at design and runtime levels;
- Quality improvement by improving predictability and conformance to specifications.

4 Conclusions and Future Work

With a total of 40 deliverables and multiple tools provided via three complementary tool sets [39], MegaM@Rt^2 aims at improving the productivity and quality of the system development and at reducing the time-to-market for complex systems, as well as to reinforce the European scientific and technological leadership and competitiveness of the European market.

The project has already delivered a significant number of research approaches, technical tools and methods spanning from system-level modeling to runtime analysis and global traceability and model management. While the results are globally evaluated as substantial, we face several open challenges towards our goal for scalable and traceable model-driven engineering applicable to a variety of industrial domains. The first phase of the project put in place the baseline methods that were assessed in the industrial settings. We demonstrated the opportunities brought by the global traceability and model management technologies resulting from research activities. In the meantime, we identified several further challenges. One of them is the need for a common runtime trace format, i.e. a shared representation for different types of runtime (meta)data. Another challenge is the need for more automated inference methods that could systematically relate these runtime traces (uniformly represented/modeled) to the corresponding system design artifacts.

Therefore, during its last period, the project plans to concentrate on those open areas by scheduling dedicated activities such as hackathons, demonstration session and workshops. We would like to engage the project community, both case study providers and technology providers to focus on a common agenda that would push the state-of-the-art further (in these areas, but also in others of interest to the project). Moreover, we plan activities to create awareness about the approaches and technologies developed in the project, which have already been adopted and endorsed by the industrial partners. Finally, an important aspect is about planning and preparing for the sustainability of the project results by creating an ecosystem for all the tools and methods composing the MegaM@Rt^2 framework.

Acknowledgement. This project has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No. 737494. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation program and from Sweden, France, Spain, Italy, Finland and Czech Republic.

References

1. Afzal, W., et al.: The MegaM@Rt2 ECSEL project: MegaModelling at runtime – scalable model-based framework for continuous development and runtime validation of complex systems. Microprocess. Microsyst. **61**, 86–95 (2018)
2. Sadovykh, A., et al.: Model-based system engineering in practice: document generation-MegaM@Rt2 project experience. In: Proceedings of the 14th Central and Eastern European Software Engineering Conference, pp. 9:1–9:6 (2018)
3. MegaMart2 - MegaModelling at runtime: MegaMart2 - MegaModelling at runtime. https://megamart2-ecsel.eu/. Accessed 25 June 2019
4. ECSEL's multi-annual strategic plan 2016. http://ec.europa.eu/research/participants/data/ref/h2020/other/legal/jtis/ecsel-multi-stratplan-2016_en.pdf. Accessed 25 June 2019
5. Sadovykh, A., et al.: A tool-supported approach for building the architecture and roadmap in MegaM@Rt2 project. In: Ciancarini, P., Mazzara, M., Messina, A., Sillitti, A., Succi, G. (eds.) SEDA 2018. AISC, vol. 925, pp. 265–274. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-14687-0_24
6. Pérez, B., Porres, I.: Reasoning about UML/OCL class diagrams using constraint logic programming and formula. Inf. Syst. **81**, 152–177 (2019)
7. Villar, E.: Model-driven analysis and design of IoT systems. In: 1st International Workshop on Embedded Software for Industrial IoT, Dresden, Germany (2018)
8. Cariou, E., Le Goaer, O., Brunschwig, L., Barbier, F.: A generic solution for weaving business code into executable models. In MODELS 2018 ACM/IEEE 21th International Conference on Model Driven Engineering Languages and Systems, Copenhagen, Denmark (2018)
9. Muttillo, V., Valente, G., Pomante, L.: Design space exploration for mixed-criticality embedded systems considering hypervisor-based SW partitions. In: 2018 21st Euromicro Conference on Digital System Design (DSD) (2018)
10. Ciambrone, D., Muttillo, V., Pomante, L., Valente, G.: HEPSIM: an ESL HW/SW co-simulator/analysis tool for heterogeneous parallel embedded systems. In: 2018 7th Mediterranean Conference on Embedded Computing (MECO) (2018)
11. Wiik, J., Ersfolk, J., Walden, M.: A contract-based approach to scheduling and verification of dynamic dataflow networks. In: 2018 16th ACM/IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE) (2018)
12. Vain, J., Truscan, D., Iqbal, J., Tsiopoulos, L.: On the benefits of using aspect-orientation in UPPAAL timed automata. In: 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS) (2017)
13. Medina, J.L., Villar, E.: Towards MARTE ++: an enhanced UML-based language to Model and Analyse Real-Time and Embedded Systems for the IoT age. Presented at the Forum on specification & Design Languages (FDL 2017), Verona, Italy (2017)
14. Arcelli, D., Cortellessa, V., Di Pompeo, D.: Automating performance antipattern detection and software refactoring in UML models. In: 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER) (2019)
15. Arcelli, D., Cortellessa, V., Di Pompeo, D., Eramo, R., Tucci, M.: Exploiting architecture/runtime model-driven traceability for performance improvement. In: 2019 IEEE International Conference on Software Architecture (ICSA) (2019)
16. Nybom, K., Ashraf, A., Porres, I.: A systematic mapping study on API documentation generation approaches. In: 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (2018)

17. Marinescu, R., Filipovikj, P., Enoiu, E.P., Larsson, J., Seceleanu, C.: An energy-aware mutation testing framework for EAST-ADL architectural models. In: 29th Nordic Workshop on Programming Theory, Turku, Finland (2018)
18. Marinescu, R., Enoiu, E., Seceleanu, C., Sundmark, D.: Automatic test generation for energy consumption of embedded systems modeled in EAST-ADL. In: 2017 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW) (2017)
19. Siavashi, F., Truscan, D., Vain, J.: Vulnerability assessment of web services with model-based mutation testing. In: 2018 IEEE International Conference on Software Quality, Reliability and Security (QRS) (2018)
20. Iqbal, J., Truscan, D., Vain, J., Porres, I.: Reconstructing timed symbolic traces from rtioco-based timed test sequences using backward-induction. In: Proceedings of the Fifth European Conference on the Engineering of Computer-Based Systems – ECBS 2017 (2017)
21. Stratis, A., Causevic, A.: A practical approach towards validating HIL simulation of a safety-critical system. In: 2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW) (2017)
22. Bergstrom, H., Enoiu, E.P.: Using timed base-choice coverage criterion for testing industrial control software. In: 2017 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW) (2017)
23. Enoiu, E., Sundmark, D., Causevic, A., Pettersson, P.: A comparative study of manual and automated testing for industrial control software. In: 2017 IEEE International Conference on Software Testing, Verification and Validation (ICST) (2017)
24. Ahmad, T., Truscan, D., Porres, I.: Identifying worst-case user scenarios for performance testing of web applications using Markov-chain workload models. Future Gener. Comput. Syst. **87**, 910–920 (2018)
25. Diaz, A., Pena, D., Villar, E.: Short and long distance marker detection technique in outdoor and indoor environments for embedded systems. In: 2017 32nd Conference on Design of Circuits and Integrated Systems (DCIS) (2017)
26. Gronback, R.: Eclipse modeling project | the eclipse foundation. https://www.eclipse.org/modeling/emf/. Accessed 25 June 2019
27. Modelio open source - UML and BPMN modeling tool. https://www.modelio.org/. Accessed 25 June 2019
28. Daniel, G., et al.: NeoEMF: a multi-database model persistence framework for very large models. Sci. Comput. Programm. **149**, 9–14 (2017)
29. Bruneliere, H., Perez, J.G., Wimmer, M., Cabot, J.: EMF views: a view mechanism for integrating heterogeneous models. In: Johannesson, P., Lee, M.L., Liddle, S.W., Opdahl, A. L., López, Ó.P. (eds.) ER 2015. LNCS, vol. 9381, pp. 317–325. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25264-3_23
30. Cicchetti, A., Di Ruscio, D., Eramo, R., Pierantonio, A.: JTL: a bidirectional and change propagating transformation language. In: Malloy, B., Staab, S., van den Brand, M. (eds.) SLE 2010. LNCS, vol. 6563, pp. 183–202. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19440-5_11
31. Desfray, P.: Model repositories at the enterprises and systems scale: the Modelio Constellation solution. In: 2015 International Conference on Information Systems Security and Privacy (ICISSP) (2015)
32. Eramo, R., Pierantonio, A., Tucci, M.: Enhancing the JTL tool for bidirectional transformations. In: Conference Companion of the 2nd International Conference on Art, Science, and Engineering of Programming – Programming 2018 Companion (2018)
33. Cortellessa, V., Eramo, R., Tucci, M.: Availability-driven architectural change propagation through bidirectional model transformations between UML and petri net models. In: 2018 IEEE International Conference on Software Architecture (ICSA) (2018)

34. Bruneliere, H., Burger, E., Cabot, J., Wimmer, M.: A feature-based survey of model view approaches. Softw. Syst. Model. **18**(3), 1931–1952 (2019)
35. Bruneliere, H., Marchand, F., Daniel, G., Cabot, J.: Towards scalable model views on heterogeneous model resources. In: ACM/IEEE 21th International Conference on Model Driven Engineering Languages and Systems (MODELS 2018), Copenhagen, Denmark, pp. 334–344 (2018)
36. Eramo, R., et al.: Model-driven design-runtime interaction in safety critical system development: an experience report. In: 15th European Conference on Modelling Foundations and Applications (ECMFA), Co-located with STAF 2019, Eindhoven, The Netherlands (2019)
37. Bruneliere, H., et al.: Model-driven engineering for design-runtime interaction in complex systems: scientific challenges and roadmap. In: Mazzara, M., Ober, I., Salaün, G. (eds.) STAF 2018. LNCS, vol. 11176, pp. 536–543. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04771-9_40
38. Deliverables - MegaMart2 - MegaModelling at Runtime. https://megamart2-ecsel.eu/deliverables/. Accessed 25 June 2019
39. MegaM@Rt2 tool box. http://toolbox.megamart2-ecsel.eu/. Accessed 25 June 2019