

What is your vision for the future of executable modeling?

(What are you currently working on? Which research questions do you try to answer? Which projects do you currently do? Which tools do you develop? Whom to contact for more information?)

Vision: Making the development of model execution tools for DSMLs easier.

Research Questions: How to provide out-of-the-box model testing, debugging and fault localization support for executable DSMLs? How effective are they compared to tailor-made tools?

Project: LEA xDSML - Language Engineering for Analyzable Executable DSMLs
<https://modelexecution.org/lea/>

Contact: Tanja Mayerhofer <mayerhofer@big.tuwien.ac.at>, Manuel Wimmer <wimmer@big.tuwien.ac.at>

Vision: Towards automated testing and generic tooling for DSMLs

RQs: How to interact with models in a unified way? What is the coverage metric of a given DSML? How to define the behavioral interface of DSMLs and use such interfaces for generic testing and debugging? How to analyse the definition of a DSML to generate relevant test suites?

Project: TETRA Box
<https://modeltransformation.net/tetrabox/>

Contact: Manuel Wimmer <wimmer@big.tuwien.ac.at>, Erwan Bousse <<mailto:merwan.bousse@ls2n.fr>>

Vision: Towards modularly constructing bespoke analysis tools

Using explicit operational, composable semantics of domain-specific modelling languages, can we directly use these as analysis tools (e.g., for performance, security, ...). This would have two potential benefits:

1. Having an explicit, inspectable behavioural semantics gives some opportunity to understand the reliability / correctness of the analysis tool

October 25, 2018

2. Modularity and robust composability will allow project teams to worry only a bout modelling the information required for the kinds of analysis relevant to them

Contact: Steffen Zschaler <szschaler@acm.org>

Vision: Try (if possible?) to have a wide adoption of MDE techniques in general before adoption of model execution in particular

MDE is used in specific developments, mainly system engineering, in big companies, but I have formed hundreds of master students who has become software engineers making standard development in Java, .Net or JavaScript now but I guess no one of them is really working with MDE techniques or tools.

Contact: Eric Cariou <eric.cariou@univ-pau.fr>

Vision: An executable model development environment that software developers would want to use

Executable modeling tools need to be actually *easier* for a developer to use than traditional software development tools. I would like to see a tool that replaces the extensive dialog boxes and property sheets common in a modeling tool with a textual language that can be used for entering and visualizing model elements at a detailed level. But this would be automatically, bidirectionally synchronized with multiple graphical views. The tooling would provide all the capabilities for code completion, refactoring, repository management, etc. of a modern IDE, but the overall language would still have all the higher-level mechanisms (e.g., data flow, state machines, timing diagrams, etc.) of modern modeling languages.

Right now I have been mostly working on building out capabilities within existing UML tools. But achieving this vision will, I think, need building new tools from the ground up. There may be enough excitement round SysML v2 tools to get new tooling funded. And perhaps these will also provide a basis for new model-based software development environments.

Contact: Ed Seidewitz ed-s@modeldriven.com

Vision: Modeling the requirements of the designer in term of execution (why does he need to execute, what are its expectation, ...) in order to build the appropriate executable model/executable modeling language.

Model execution (also program execution on test platform instead of deployment platform) is successful when it allows to improve the productivity in system/software development (it must thus be cost effective in its development and use).

October 25, 2018

Contact: Marc Pantel <marc.pantel@n7.fr>

Vision: wider adoption of executable and translatable modeling in OSS

Working on: BridgePoint (xtuml.org)

Contact: <mailto:support@onefact.net>

Vision: teaching materials

I am working on materials to be used in executable modeling training that are open and widely accessible. It seems that explaining verbally and reading about modeling can be less efficient than step-by-step exercises.

Cortland Starrett <cortland.starrett@onefact.net>

Vision: Integrating modeling and programming

Research Questions: Program code is often a good formalism to describe behaviour. How to integrate program code with arbitrary behavioural and structural models? The focus in this project is currently shifting to: how can models with integrated program code be executed, monitored, debugged in combination, and how can execution traces be explored?

Project: Codeling (<https://codeling.de>)

Contact: Marco Konersmann <marco.konersmann@paluno.uni-due.de>

Vision: Identifying existing programming language as a behavioural modeling language

It is obvious that software engineers need time and efforts to learn and use new languages even if they love to do them, and thus introducing new languages to each new domain causes them exhausted. I am considering an approach that reuses the syntax of an existing language by redefining their semantics such that have the same meaning as the original ones but different specifications in a domain. For example, the use of hash tables can be regarded as an access to object storage service or key-value store service in the domain of cloud computing, and we could model a behaviour using the services simply as the manipulation of a hash table. In the approach, we just need model transformation rules giving domain specifications to each AST node, and the developers' effort for behavioural modeling must be decreased compared to

October 25, 2018

learning a new behavioural modeling language; they only need their programs and few configurations to a domain for executing the programs.
I am tackling the approach in JavaScript AST by applying model transformation techniques.

Contact: Kosaku Kimura <kimura.kosaku@jp.fujitsu.com>

Vision: Validating models against real operational data via model execution

We have done some work on validating belief test ready models and supporting their evolution via model execution. The work has been published in:

<https://www.sciencedirect.com/science/article/pii/S0950584917302161>

The work was done in the context of the U-Test project (<http://u-test.eu/>).

Contact: Tao Yue (tao@simula.no) and Shaukat Ali (shaukat@simula.no)

Vision: Combining software IDEs and modeling tools

So, as stated on the workshop my opinion is that the modelling community is doing a mistake trying to build replacements for the currently successful programming languages and development environments. Instead we need to focus on where modeling provides value and combine modelling with development languages and environments.

Contact: <anders.ek@hcl.com>

For an example see <https://www.devops-community.com/realtime-software-tooling-rtist.html>

Vision: Bridge the gap between industrial activities and energy efficiency

I'm working on a DSL to let experts model industrial activities (from farms to tertiary activities): time of use, variability etc... and energy related appliances (solar panels, machines, batteries). The objective is to explore energy efficiency and how to improve it.

Contact: alexandre.rio@irisa.fr

Vision: better models, better tools, better systems for robotics

Composable Models and Software for Robotics Systems

October 25, 2018

RobMoSys is *the* place to talk about all aspects of models in the context of robotics software systems. So be invited to and highly welcome on our discourse platform to discuss with the robotics community on model-driven approaches in robotics.

- executable models are a very relevant aspect in robotics as robots finally need to act in real world. Without having your model level of abstraction be grounded in the next lower / more concrete level, you will not end up in a robot acting in real world.
- we strongly focus on composition, compositionality, composability as we need to come up with re-use of commodity software components etc. to make complex software-intensive robot systems more affordable, changeable, etc.

See EU H2020 project RobMoSys

- <https://robmosys.eu>
- <https://robmosys.eu/wiki>
- <https://discourse.robmosys.eu>

Contact: Christian Schlegel, schlegel@hs-ulm.de

Vision: Modular interface between model analysis tools and model execution engines

Provide general interfaces between model analysis tools (simulator, debuggers, model-checkers) and model execution engines.

It remains a time-consuming, complex, ad-hoc task to support debugging, simulation or any other analysis/diagnosis feature for each new DSL.

Maybe a generic protocol can be developed and standardized to make model analysis tools communicate with model execution engines in a generic way (independently on the way models have been modeled).

See the Debug Adapter Protocol (DAP) or the Language Server Protocol (LSP) that offers this kind of interface for debugging and languages respectively.

I am currently working on a model interpreter that can execute models but that can also be used during the diagnosis phase. Having such a modular interface for this model interpreter is one of the challenges me and my team are working on.

Contact: Valentin Besnard <valentin.besnard@eseo.fr>