Reiner Hähnle

Technische Universität Darmstadt

HATS Annual Review Meeting 2012

http://www.hats-project.eu
### HATS: Highly Adaptable & Trustworthy Software Using Formal Models

- **FP7 FET focused call** *Forever Yours*
- **Project started** 1 March 2009, 48 months runtime
- **Integrated Project, academically driven**
- **10 academic partners, 2 industrial research, 1 SME**
  - Coordinator moved to TU Darmstadt on 1 Sep, 2012
  - New beneficiary TUD
- **8 countries**
- **805 PM, EC contribution 5,64 M€ over 48 months**
- **Associated with FP7 Coordination Action: EternalS**
  - Trustworthy Eternal Systems via Evolving Software, Data and Knowledge
The **highlighted** people are present

<table>
<thead>
<tr>
<th>Name</th>
<th>Institution</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hähnle, Bubel</td>
<td>Technische Univ. Darmstadt <em>(Sci. Coor.)</em></td>
<td>DE</td>
</tr>
<tr>
<td>Ahrendt, Waborg</td>
<td>Chalmers Tekniska Högskola <em>(Adm. Coor.)</em></td>
<td>SE</td>
</tr>
<tr>
<td>Johnsen, Schlatte</td>
<td>Universitetet i Oslo</td>
<td>NO</td>
</tr>
<tr>
<td>Dam, Gurov, Meinke</td>
<td>Kungliga Tekniska Högskolan</td>
<td>SE</td>
</tr>
<tr>
<td>Albert, Barthe, Puebla</td>
<td>Universidad Politécnica de Madrid/IMDEA</td>
<td>ES</td>
</tr>
<tr>
<td>Poetzsch-Heffter</td>
<td>University of Kaiserslautern</td>
<td>DE</td>
</tr>
<tr>
<td>Sangiorgi, Bravetti</td>
<td>Università di Bologna</td>
<td>IT</td>
</tr>
<tr>
<td>De Boer</td>
<td>Centrum voor Wiskunde en Informatica</td>
<td>NL</td>
</tr>
<tr>
<td>Hagalisletto, Østvold</td>
<td>Norsk Regnesentral</td>
<td>NO</td>
</tr>
<tr>
<td>Diakov, Wong</td>
<td>Fredhopper</td>
<td>NL</td>
</tr>
<tr>
<td>Villela, Arif</td>
<td>Fraunhofer ISE</td>
<td>DE</td>
</tr>
<tr>
<td>Clarke, Proença</td>
<td>Katholieke Universiteit Leuven</td>
<td>BE</td>
</tr>
<tr>
<td>Uustalu, Laud</td>
<td>Institute of Cybernetics</td>
<td>EE</td>
</tr>
</tbody>
</table>
In a nutshell, we ... develop a tool-supported formal modeling language (ABS) for the design, analysis, and implementation of highly adaptable software systems characterized by a high expectations on trustworthiness for target software systems that are ... concurrent, distributed, object-oriented, built from components, adaptable (variability, evolvability), hence reusable.

Main focus: Software Product Line Engineering
Motivation

Why formal?

- informal notations can’t describe software behavior with rigor: concurrency, modularity, correctness, security, resources . . .
- formalization $\Rightarrow$ more advanced tools
  - more complex products
  - higher automation: cost-efficiency

Why adaptable?

- software rich in features and variants, many deployment scenarios
- changing requirements (rapid technological/market pace)
- evolution of software in unanticipated directions
- language-supported adaptability is a key to successful reuse
Motivation

Why formal?

▶ informal notations can’t describe software **behavior** with rigor:
  concurrency, modularity, correctness, security, resources . . .

▶ formalization $\Rightarrow$ more advanced tools
  • more complex products
  • higher automation: cost-efficiency

Why adaptable?

▶ software rich in features and variants, many deployment scenarios

▶ changing requirements (rapid technological/market pace)

▶ evolution of software in unanticipated directions

▶ language-supported adaptability is a key to successful **reuse**
Mind the Gap!

Design-oriented, architectural, structural
UML, FDL, etc.

Implementation level
JML, SPEC#, etc.

Minimalistic foundational
\(\pi\)-calculus, ambient c., etc.
Mind the Gap!

Design-oriented, architectural, structural UML, FDL, etc.

+ executability

Abstract Behavioural Specification ABS

+ verifiability

Implementation level JML, SPEC#, etc.

+ usability

Minimalistic foundational $\pi$-calculus, ambient c., etc.
How?

A tool-supported formal method for building highly adaptable and trustworthy software
How?

A tool-supported formal method for building highly adaptable and trustworthy software

Main ingredients

1. Executable, **formal** modeling language for adaptable software: *Abstract Behavioral Specification* (ABS) language

2. **Tool suite** for ABS/executable code analysis & development:
   - **Analytic** functional/behavioral verification, resource analysis, feature consistency, RAC, types, TCG, visualization
   - **Generative** code generation, model mining, monitor inlining, . . .

Develop methods **in tandem** with ABS to ensure feasibility

3. Methodological and technological **framework** integrating HATS tool architecture and ABS language
Important Project Principles (I)

Ensuring relevance

- Apply to empirically highly successful development method: **Software product line engineering (PLE)**
- Thorough requirements analysis, continuous evaluation

![Diagram of project process]

- **Feature Model** → **Family Engineering** → **Product Line Artefacts Base** → **Application Engineering** → **Product**
Important Project Principles (II)

Feasibility: ensure that analysis methods scale up

Develop analysis methods in tandem with ABS language

Incrementality
  ▶ Delta modeling, delta specification, delta verification

Compositionality
  ▶ Concurrency model
  ▶ Proof systems

Automation
  ▶ Type systems for Near/Far-analysis, deadlocks, safe products
  ▶ Resource analysis, test case generation

Ease of Usage
  ▶ Integrate into design methodology/workflow
  ▶ Integrate into standard IDE (Eclipse)
Important Project Principles (III)

Early evaluation

- Develop Core ABS first

Delta Modeling Languages:
- µTVL
- DML
- CL
- PSL

Component Model Deployment Components:
- Real-Time ABS
- Local Contracts, Assertions
- Syntactic Modules
- Asynchronous Communication
- Concurrent Object Groups
- Imperative Language
- Object Model
- Pure Functional Programs
- Algebraic Data Types
Important Project Principles (III)

**Early evaluation**
- Develop Core ABS first
- Layered language design

**Delta Modeling**
- Languages: \( \mu \text{TVL, DML, CL, PSL} \)
- Component Model

**Deployment Components:**
- Real-Time ABS

**Local Contracts, Assertions**
- Syntactic Modules
- Asynchronous Communication
- Concurrent Object Groups
- Imperative Language
- Object Model
- Pure Functional Programs
- Algebraic Data Types
Important Project Principles (III)

**Early evaluation**
- Develop Core ABS first
- Layered language design
- Provide tools early

Diagram:
- Core AST
  - Name Resolution
  - Resolved AST
  - Type Checker
  - Type-Checked AST
    - Maude Back End
      - Maude Files
      - Maude VM
    - Java Back End
      - Java Files
      - Java VM
    - Core ABS code gen.
      - Core ABS Files

Languages:
- µTVL
- DML
- CL
- PSL

Deployment Components:
- Real-Time ABS
- Core AST
- Name Resolution
- Resolved AST
- Type Checker
- Type-Checked AST
- ABS IDE
- Core ABS IDE
The Main Innovations of HATS

ABS—a formal, executable, abstract, behavioral modeling language
- Cutting-edge research on modeling of concurrent, OO systems
- Combines state-of-art in verification, concurrency, specification, and programming languages communities
- Tailored to model feature-rich and evolvable systems

Scalable technologies developed in tandem with ABS
- Incremental, compositional
- Analytic as well as generative technologies

Formalization of PLE-based development as main application
- Leveraging formal methods tools to Product Line Engineering
- Define FM-based development methodology for PLE
Vision: A Single-Source Technology for Highly Adaptive, Concurrent Software Systems

- UML class diagram
- Object diag.
- Sequence diag.
- Architect. lang.
- Feature descr. lang.
- Bytecode
- Maude
- Erlang
- Petri net
- ABS

Maude
Scala
Vision: A Model-Centric Development Method for PLE

Product Line Models expressed in HATS ABS with uniform formal semantics

Consistency analysis, correctness of reuse, family visualization, test case generation, validation, verification, family evolution, rapid prototyping, code generation, product visualization, test case generation, validation, verification, product evolution

Application Engineering

[Schaefer & Hähnle, IEEE Computer, Feb. 2011]
Work Organisation in HATS

WP1: Framework
language design, methodology, tool infrastructure, integration

WP2: Variability
anticipated change

WP3: Evolvability
unanticipated change

WP4: Trustworthiness
cross-cutting qualities

WP5: Validation

modeling
analysis
<table>
<thead>
<tr>
<th>Del. no.</th>
<th>Deliverable name</th>
<th>WP no.</th>
<th>Lead beneficiary</th>
<th>Nature</th>
<th>Dissemi- nation level</th>
<th>Delivered</th>
<th>Actual/ Forecast</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>D2.1</td>
<td>Configuration Deployment</td>
<td>2</td>
<td>UIO</td>
<td>report</td>
<td>public</td>
<td>✔</td>
<td>35/35</td>
<td>shifted from PM30</td>
</tr>
<tr>
<td>D2.3</td>
<td>Testing, Debugging and Visualization</td>
<td>2</td>
<td>TUD</td>
<td>report</td>
<td>public</td>
<td>✔</td>
<td>36/36</td>
<td></td>
</tr>
<tr>
<td>D2.4</td>
<td>Types for Variability</td>
<td>2</td>
<td>BOL</td>
<td>report</td>
<td>public</td>
<td>✔</td>
<td>36/36</td>
<td></td>
</tr>
<tr>
<td>D3.2</td>
<td>Model mining</td>
<td>3</td>
<td>NR</td>
<td>report</td>
<td>public</td>
<td>✔</td>
<td>36/36</td>
<td></td>
</tr>
<tr>
<td>D4.1</td>
<td>Security</td>
<td>4</td>
<td>IMDEA</td>
<td>report</td>
<td>public</td>
<td>✔</td>
<td>36/36</td>
<td></td>
</tr>
<tr>
<td>D5.3</td>
<td>Evaluation of modeling</td>
<td>5</td>
<td>CWI</td>
<td>report</td>
<td>public</td>
<td>✔</td>
<td>36/36</td>
<td>validated M2</td>
</tr>
</tbody>
</table>
## Milestones

<table>
<thead>
<tr>
<th>Milestone no.</th>
<th>Milestone name</th>
<th>WP no.</th>
<th>Lead beneficiary</th>
<th>Achieved</th>
<th>Actual/Forecast</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>Core language and methodology</td>
<td>1</td>
<td>UIO</td>
<td>✓</td>
<td>12/12</td>
<td>D1.1a, D1.1b Validated in D5.2</td>
</tr>
<tr>
<td>M2</td>
<td>Modeling capabilities</td>
<td>1,2,3</td>
<td>UKL</td>
<td>✓</td>
<td>24/24</td>
<td>D1.2, D2.2b, D3.1b Validated in D5.3</td>
</tr>
<tr>
<td>M3</td>
<td>Analysis and development methods</td>
<td>2,3,4</td>
<td>TUD</td>
<td>✓</td>
<td>36/36</td>
<td>D2.1, D2.3, D2.4, D2.5, D3.2, D3.4, D4.1, D4.2 To be validated in D5.4</td>
</tr>
<tr>
<td>Nr</td>
<td>Name</td>
<td>Lead</td>
<td>Start</td>
<td>Deliv.</td>
<td>Comment</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>----------------------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
<td>----------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Analysis</td>
<td>TUD</td>
<td>PM25</td>
<td>—</td>
<td>Early start Milestone M4 formerly CTH</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>System derivation and code generation</td>
<td>IoC</td>
<td>PM25</td>
<td>—</td>
<td>Milestone M4</td>
<td></td>
</tr>
<tr>
<td>1.5</td>
<td>Integrated tool platform</td>
<td>NR</td>
<td>PM25</td>
<td>—</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td>A configurable deployment architecture</td>
<td>UIO</td>
<td>PM7</td>
<td>2.1</td>
<td>Extended Milestone M3 Finished</td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>Testing, debugging, and visualization</td>
<td>TUD</td>
<td>PM19</td>
<td>2.3</td>
<td>Milestone M3 Finished formerly CTH</td>
<td></td>
</tr>
<tr>
<td>2.4</td>
<td>Types for variability</td>
<td>BOL</td>
<td>PM13</td>
<td>2.4</td>
<td>Milestone M3 Finished</td>
<td></td>
</tr>
<tr>
<td>2.6</td>
<td>Refinement and abstraction</td>
<td>UKL</td>
<td>PM25</td>
<td>—</td>
<td>Milestone M4</td>
<td></td>
</tr>
<tr>
<td>Nr</td>
<td>Name</td>
<td>Lead</td>
<td>Start</td>
<td>Deliv.</td>
<td>Comment</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>---------------------------------------------------</td>
<td>------</td>
<td>-------</td>
<td>--------</td>
<td>--------------------------------------</td>
<td></td>
</tr>
<tr>
<td>3.2</td>
<td>Model mining</td>
<td>NR</td>
<td>PM19</td>
<td>3.2</td>
<td>Milestone M3 Finished</td>
<td></td>
</tr>
<tr>
<td>3.3</td>
<td>Hybrid analysis for evolvability</td>
<td>KUL</td>
<td>PM25</td>
<td>—</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5</td>
<td>Autonomously evolving systems</td>
<td>KTH</td>
<td>PM25</td>
<td>—</td>
<td>Early start</td>
<td></td>
</tr>
<tr>
<td>4.1</td>
<td>Security</td>
<td>IMDEA</td>
<td>PM13</td>
<td>4.1</td>
<td>Milestone M3 Finished</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Correctness</td>
<td>BOL</td>
<td>PM25</td>
<td>—</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td>Auto configuration and quality variability</td>
<td>FRG</td>
<td>PM31</td>
<td>—</td>
<td>Milestone M4</td>
<td></td>
</tr>
<tr>
<td>5.3</td>
<td>Evaluation of modeling</td>
<td>CWI</td>
<td>PM19</td>
<td>5.3</td>
<td>Milestone M2 Finished</td>
<td></td>
</tr>
<tr>
<td>5.4</td>
<td>Evaluation of tool and techniques</td>
<td>FRH</td>
<td>PM31</td>
<td>—</td>
<td>Milestone M4</td>
<td></td>
</tr>
</tbody>
</table>

15 work tasks finished, 9 continue, **all** tasks started
# Third Year Objectives & Results: Milestone 3

<table>
<thead>
<tr>
<th>Development Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ Delta modeling workflow based on abstract delta modeling (T5.3)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Analysis Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ Near/far location type analysis (T1.3), deadlock analysis (T4.3)</td>
</tr>
<tr>
<td>▶ Type system for checking type-safety of delta models (T2.4)</td>
</tr>
<tr>
<td>▶ Glass box test case generator and ABSUnit framework (T2.3)</td>
</tr>
<tr>
<td>▶ Model checker for products of a SW product line (T1.3)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Generative Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ Model mining from code, traces, product descriptions (T3.2)</td>
</tr>
<tr>
<td>▶ Automatic construction of crypto protocol implementation (T4.1)</td>
</tr>
<tr>
<td>▶ Scala backend for ABS (T1.4)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Security Policies</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ Information-flow type system for core ABS (T4.1)</td>
</tr>
<tr>
<td>▶ Logic-based and dynamic enforcement of security policies (T4.1)</td>
</tr>
</tbody>
</table>

Previous year: behavioral verification (T2.5), resource analysis (T4.2)
Dynamic Aspects, Evolvability

- **Deployment components** to model low-level notions (T2.1)
  - Schedulers, load, bandwidth
  - Real-time ABS
- **Abstract failure model** and type system (T2.1, T2.4)
- **ABS component model** (T2.1, T3.3)
- **ABS-NET**: semantics for network-aware runtime ABS (T3.5)

Evaluate Modeling

- **Fredhopper Access Server replication system** (T5.3)
  - Modeled with Full ABS as basis for validation of M2
- Some models in S.P.L.O.T. feature model repository (T5.3)
Implementation of Recommendations from Reviewers

The ABS language occupies a "niche" by focusing on an intermediate level between abstract models and code. The project is encouraged, but it should take care that the links to the more "classical" levels of abstraction are well studied.

- Task 2.6 Refinement and Abstraction: Correct refinements to/from ABS artefacts, including trace semantics, timed automata, and rule-based compilation.
- Task 3.2 Model Mining: Relate ABS to automata-based abstractions and to the Java Message Service.
- Task 3.3 Evolvability for Hybrid Systems: A suitable notion of runtime component for ABS.

Continue/improve successful development of the HATS tool suite:

- Task 1.5 Integrated Tool Platform: See demo in next session (task started 6 months earlier).

Reiner Hähnle
HATS Project Overview and Introduction
Annual Review Meeting 2012
21 / 26
The ABS language occupies a [...] “niche” by focusing on an intermediate level between abstract models and code. The project is encouraged [...], but it should take care that the links to the more “classical” levels of abstraction are well studied.

- **Task 2.6 Refinement and Abstraction:**
  Correct refinements to/from ABS artefacts, including trace semantics, timed automata, and rule-based compilation
- **Task 3.2 Model Mining:**
  Relate ABS to automata-based abstractions and to the Java Message Service
- **Task 3.3 Evolvability for Hybrid Systems:**
  A suitable notion of runtime component for ABS
The ABS language occupies a [...] “niche” by focussing on an intermediate level between abstract models and code. The project is encouraged [...], but it should take care that the links to the more “classical” levels of abstraction are well studied ...

- **Task 2.6 Refinement and Abstraction:**
  Correct refinements to/from ABS artefacts, including trace semantics, timed automata, and rule-based compilation

- **Task 3.2 Model Mining:**
  Relate ABS to automata-based abstractions and to the Java Message Service

- **Task 3.3 Evolvability for Hybrid Systems:**
  A suitable notion of runtime component for ABS

Continue/improve successful development of the HATS tool suite

- **Task 1.5 Integrated Tool Platform:**
  See demo in next session (task started 6 months earlier)
In addition to the classification of the verification properties, the roles and relevance of these properties for product line engineering should be evaluated and a systematic approach for the use of verification properties should be developed

- Task 2.4 identified type-safety of products as central
- Extended to dynamic code updates in Task 3.3
- Task 4.3 looks systematically at correctness properties
- Evaluation of analysis tools: Task/Deliverable 5.4 PM 48

Clarify the advantages/drawbacks of Delta programming [. . . ]; because of the decision for Delta modelling, the project should focus the work of the remaining tasks of WP2 on this technique

- Delta modeling was concretized into Delta Modeling Workflow process, defined and evaluated in Deliverable D5.3
- Evaluation of delta modeling to be continued in Task 5.4
- Delta modeling part of full ABS, all subsequent tasks use it
Continue to take integration seriously. Make an effort to increase joint publications which involve different project partners.

- Of 46 publications in 3rd project year, no less than 13 (ca. 30%) involved more than one project partner

The exploitation plan is well under way; however, the sustainability of tools beyond the end of the project deserves more attention.

- Some new EC/national project proposals based on ABS underway
- TUD, UIO and UKL committed to maintain ABS tool chain beyond HATS

The new Estonian partner should be effectively involved into the project work and present its results at the next review.

- In 3rd period IoC participated actively in Tasks 1.4 (lead), Task 2.4, Task 4.1, and Task 4.3
- Co-authorship of D4.1
- Today, progress report of T1.4, mentioned in T4.1 and T4.3
- Organization of this review meeting, 5 publications
Although not required by the DoW, we suggest studying the extension of the COSTA framework to adaptable systems [...] • Will be addressed in ongoing Task 4.4 Auto Configuration and Quality Variability, where UPM plans to develop incremental resource analysis in COSTABS

[... ] The reviewers recommend [...] to take appropriate action for ensuring a continuous quality of the research in the areas where Ina Schaefer was leading the work.

• Ina continues to contribute towards the project results:
  • Co-authorship of Deliverables D2.4 and D3.2
  • Five joint publications within HATS during 3rd period
  • Tenured post (TU Braunschweig) basis for continued collaboration
• Ina’s expertise successfully transferred to TUD, BOL, KUL, KTH, UIO

Although the quality of the deliverables was considerably better than in year 1, we recommend the following improvements: ... • All recommendations have been implemented
<table>
<thead>
<tr>
<th>Work Package</th>
<th>WP Leader</th>
<th>Presenters (Task Leaders)</th>
</tr>
</thead>
<tbody>
<tr>
<td>WP1 Framework, Demo</td>
<td>UKL</td>
<td>Østvold (T1.5)</td>
</tr>
<tr>
<td></td>
<td>Poetzsch-Heffter</td>
<td>Schlatte (demo, T1.5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bubel (T1.3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uustalu (T1.4)</td>
</tr>
<tr>
<td>WP2 Variability</td>
<td>UIO</td>
<td>Johnsen (T2.1)</td>
</tr>
<tr>
<td></td>
<td>Johnsen</td>
<td>Bubel (T2.3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sangiorgi (T2.4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Poetzsch-Heffter (T2.6)</td>
</tr>
<tr>
<td>WP3 Evolvability</td>
<td>KTH</td>
<td>Østvold (T3.2)</td>
</tr>
<tr>
<td></td>
<td>Dam</td>
<td>Proença (T3.3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dam (T3.5)</td>
</tr>
<tr>
<td>WP4 Trustworthiness</td>
<td>UPM/IMDEA</td>
<td>Barthe (T4.1)</td>
</tr>
<tr>
<td></td>
<td>Albert</td>
<td>Bravetti (T4.3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Albert (T4.4)</td>
</tr>
<tr>
<td>Work Package</td>
<td>WP Leader</td>
<td>Presenters (Task Leaders)</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>WP5 Validation</td>
<td>FRH</td>
<td>de Boer (T5.3)</td>
</tr>
<tr>
<td></td>
<td>Wong</td>
<td>Wong (demo, T5.4)</td>
</tr>
<tr>
<td>WP6 Dissemination and Training</td>
<td>FRG (Villela)</td>
<td>Arif (T6.1, T6.2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sangiorgi (T6.3)</td>
</tr>
<tr>
<td>WP7 Management</td>
<td>CTH (admin)</td>
<td>Hähnle (T7.1)</td>
</tr>
<tr>
<td></td>
<td>TUD (scientific)</td>
<td>Waborg (T7.2)</td>
</tr>
<tr>
<td></td>
<td>Waborg, Hähnle</td>
<td></td>
</tr>
</tbody>
</table>