Quality Assurance for Fault-Operational Cyber-Physical Systems

Franz Wotawa

Technische Universität Graz

Institute for Software Technology

CD Lab for Quality Assurance Methodologies for Cyber-Physical Systems

wotawa@ist.tugraz.at

QAMCAS

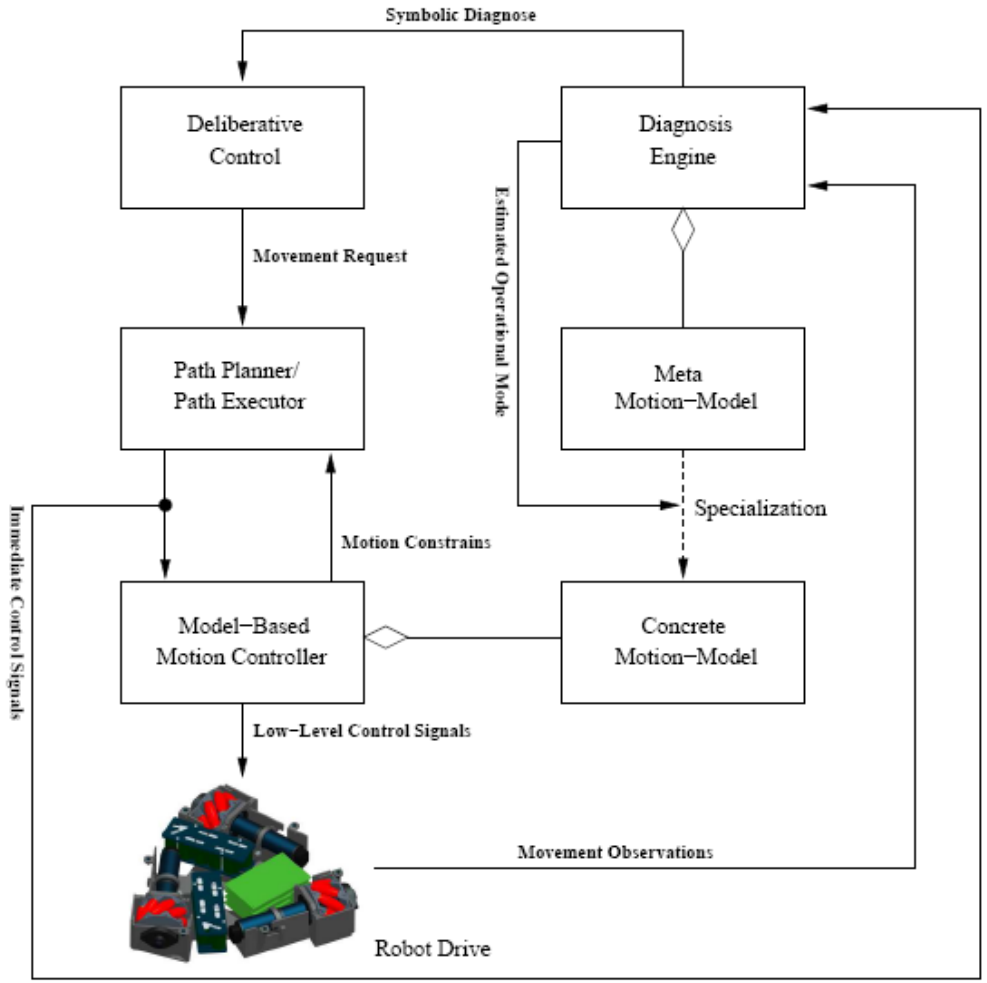
Early work (focusing on MBR)

❶

Automated compensation of HW faults

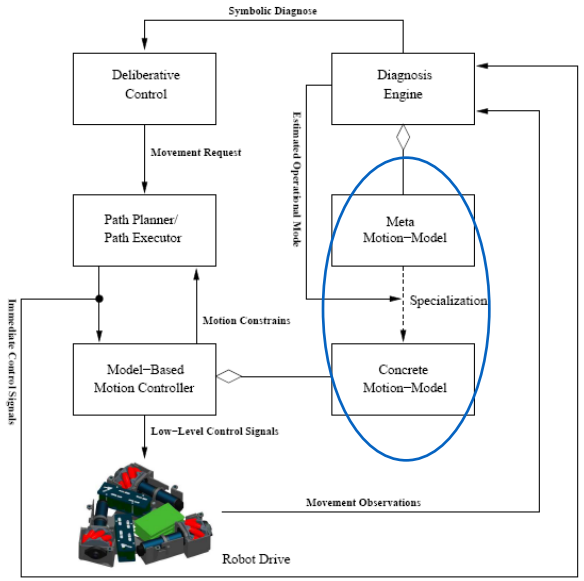
- Joint work with Michael Hofbaur, Johannes Köb, Gerald Steinbauer, Jörg Weber
- **Publications**:
  - M. Hofbaur, J. Köb, G. Steinbauer, and F. Wotawa. *Improving Robustness of mobile Robots using Model-based Reasoning*. Journal of Intelligent and Robotic Systems , Springer, Vol. 48(1), Jan, 2007.
  - G. Steinbauer, F. Wotawa. *Robust Plan Execution Using Model-Based Reasoning. Advanced Robotics* , 23 (10), pp. 1315 - 1326, 2009.
  - J. Weber and F. Wotawa. *Diagnosis and repair of dependent failures in the control system of a mobile autonomous robot*. Applied intelligence . 36 (4), pp. 511–528, 2012.

System architecture



Modeling

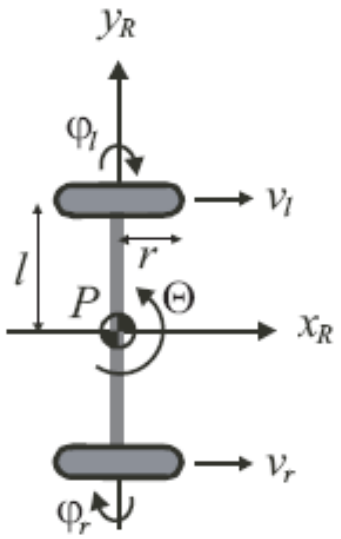
- Hybrid systems
  - Continuous behaviors that are interleaved with discrete changes
  - Physical entities of a system: discretely and continuously valued variables
- *Hybrid estimation*
  - Hybrid automaton model for the system
    → *Probabilistic hybrid automata (PHA)*



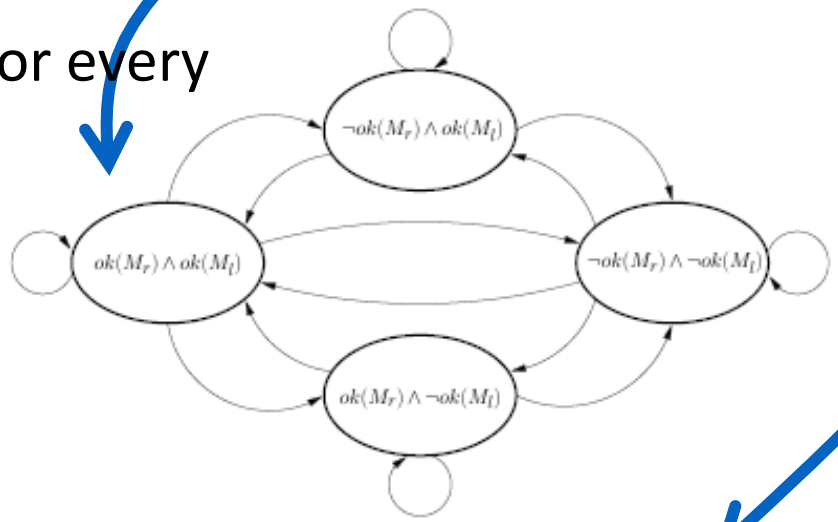
Example PHA: differential drive

- Nominal mode:

$$\begin{bmatrix} \dot{\omega}_r \\ \dot{\omega}_l \end{bmatrix} = \begin{bmatrix} 1/\tau & 0 \\ 0 & 1/\tau \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_r \\ u_l \end{bmatrix} + \mathbf{w}_x$$

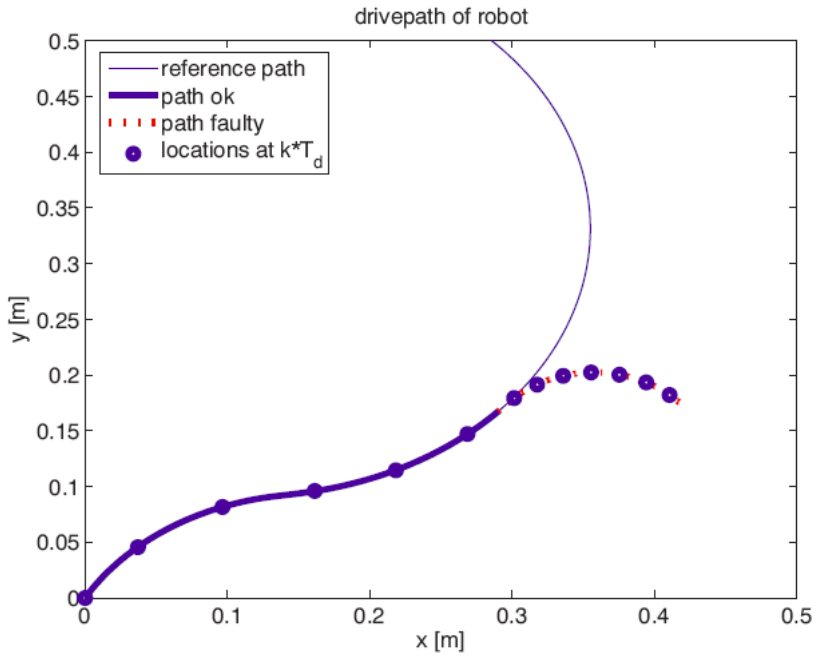


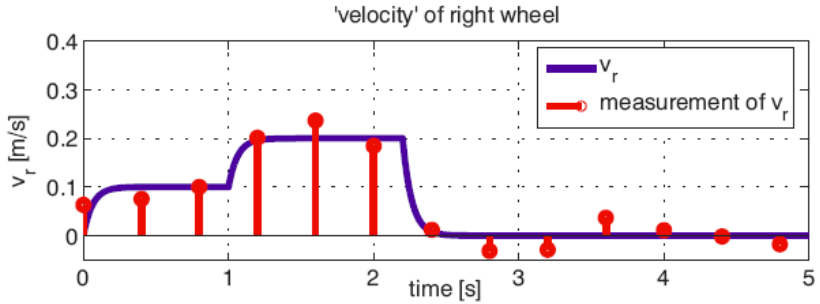
Kinematics equations for every state

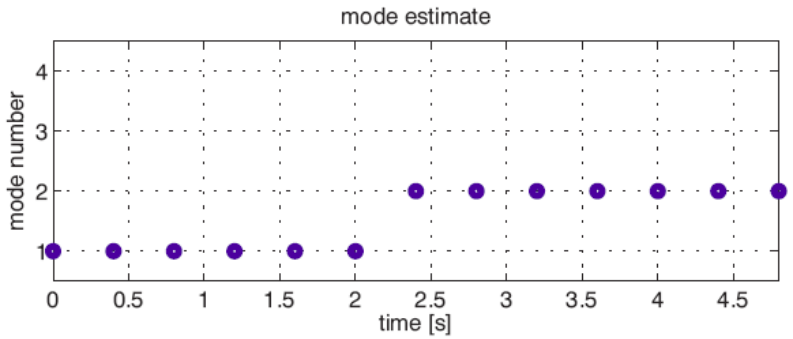


$$\mathbf{y}_c = \begin{bmatrix} -r & 0 \\ 0 & r \\ -r/2 & r/2 \\ -1/2\,l & -1/2\,l \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} + \mathbf{w}_y \,.$$

Hybrid Estimation - Example

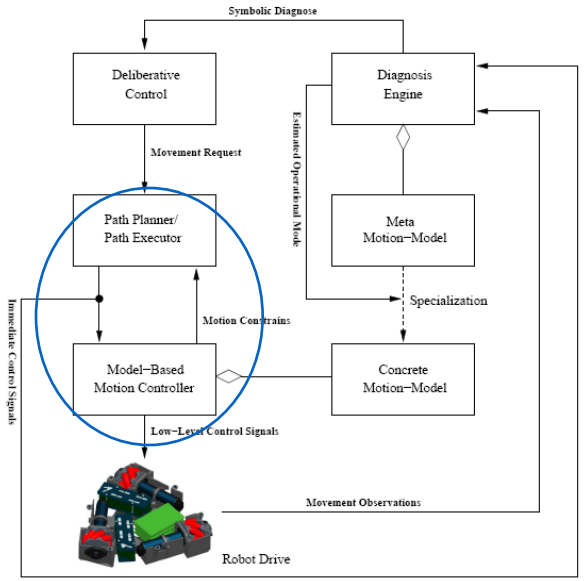






Model-based control

- After identifying the current (discrete) mode, the control model has to be adapted.
- Control converts motion commands from path planner to control signals for motors
- Use model of motors and combine them to form a model of the whole drive



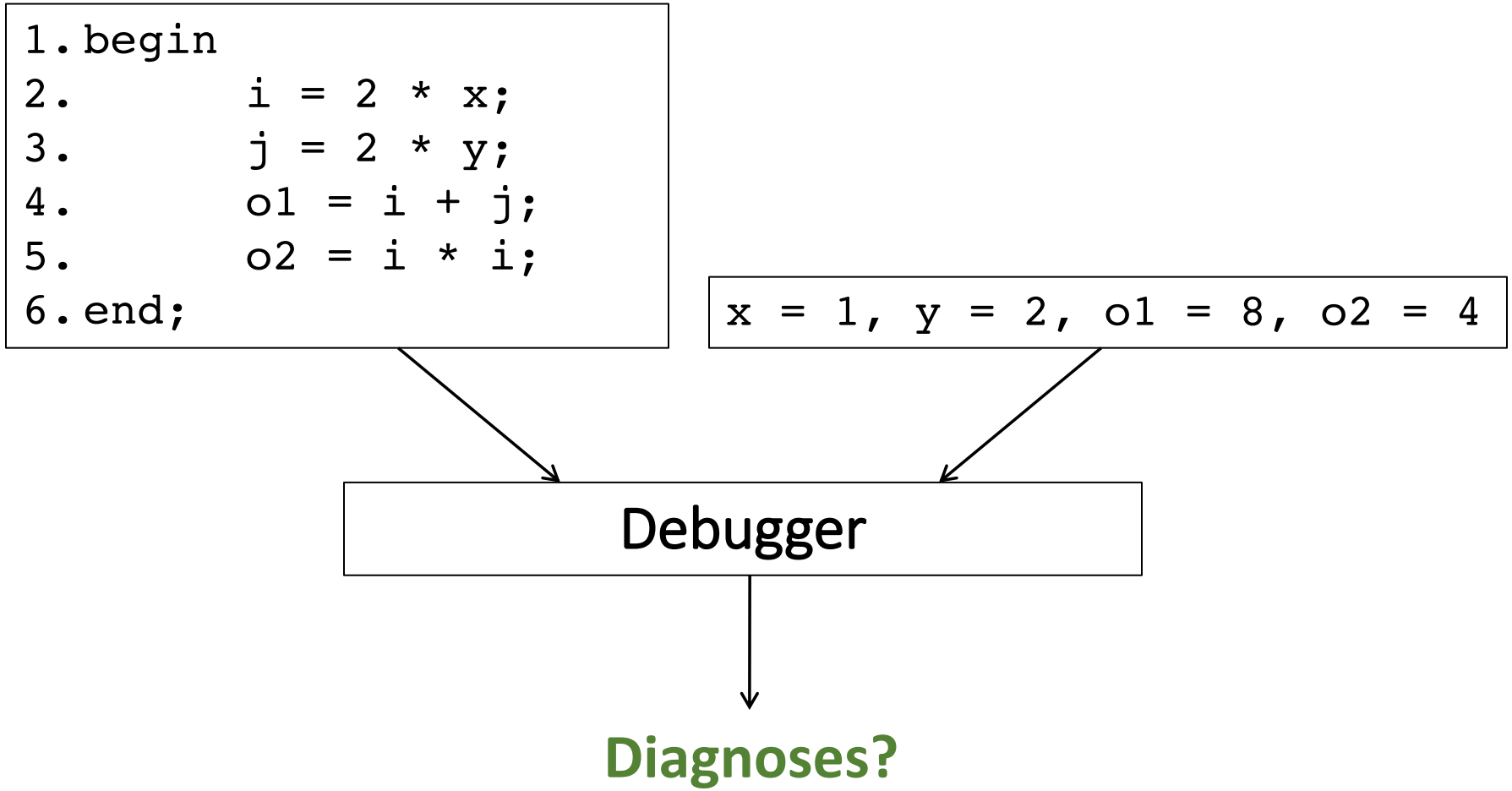

Eye-Catcher
Task

❷

Fault localization in SW

- Joint work with Markus Stumptner, Wolfgang Mayer, Dominik Wieland, Rui Abreu, Eric Wong, Birgit Hofer, Iulia Nica, Mihai Nica, Bernhard Peischl, Daniel Köb, Patrick Koch, Dietmar Jannach, Ingo Pill, Konstantin Schekotihin,…
- Publications:
  - Birgit Hofer and Andrea H¨ofler and Franz Wotawa. Combining Models for Improved Fault Localization in Spreadsheets. IEEE Transactions on Reliability, Vol. 66(1), pp. 38–53, DOI: 10.1109/TR.2016.2632151 , March, 2017.
  - W. Eric Wong, Ruizhi Gao, Yihao Li, Rui Abreu, and Franz Wotawa A Survey on Software Fault Localization. IEEE Transactions on Software Engineering, 2016. doi: 10.1109/TSE.2016.2521368
  - R. Abreu, B. Hofer, A. Perez, and F. Wotawa. On the empirical evaluation of similarity coefficients for spreadsheet fault localization. Automated Software Engineering, Vol. 22(1):47–74, 2015
  - R. Abreu, B. Hofer, A. Perez, and F. Wotawa. Using Constraints to Diagnose Faulty Spreadsheets. Software Quality Journal, pp. 126, 2014.
  - D. Jannach, T. Schmitz, B. Hofer, and F. Wotawa. Avoiding, Finding and Fixing Spreadsheet Errors – A Survey of Automated Approaches for Spreadsheet QA. Journal of of Systems and Software, Elsevier, 2014.

Debugging – A (very) short intro



Debugging using constraints

```
1. begin
2.        i = 2 * x;
3.        j = 2 * y;
4.        o1 = i + j;
5.        o2 = i * i;
6. end;
```

$Ab(2) \vee i = 2 * x;$
$Ab(3) \vee j = 2 * y;$
$Ab(4) \vee o1 = i + j;$
$Ab(5) \vee o2 = i * i;$

```
x = 1, y = 2, o1 = 8, o2 = 4
```

$x = 1$
$y = 2$
$o1 = 8$
$o2 = 4$

Programm execution

Constraint solving / equation solving

Finding bugs using constraints

$Ab(2) \vee$ ~~$i = 2 * x;$~~
$Ab(3) \vee$ ~~$j = 2 * y;$~~
$Ab(4) \vee$ $o1 = i + j;$
$Ab(5) \vee$ $o2 = i * i;$

$x = 1$
$y = 2$
$o1 = 8$
$o2 = 4$

$Ab(2) \wedge \neg Ab(3) \wedge \neg Ab(4) \wedge \neg Ab(5)$

j = 2 * 2 = 4
o1 = i + j = 8 = i + 4 → i = 4
o2 = 4 = i * i = 4 * 4 → **FAIL!!!!**

$\neg Ab(2) \wedge Ab(3) \wedge \neg Ab(4) \wedge \neg Ab(5)$

i = 2 * 1 = 2
o1 = 8 = 2 + j → j = 6
o2 = 4 = i * i = 2 * 2

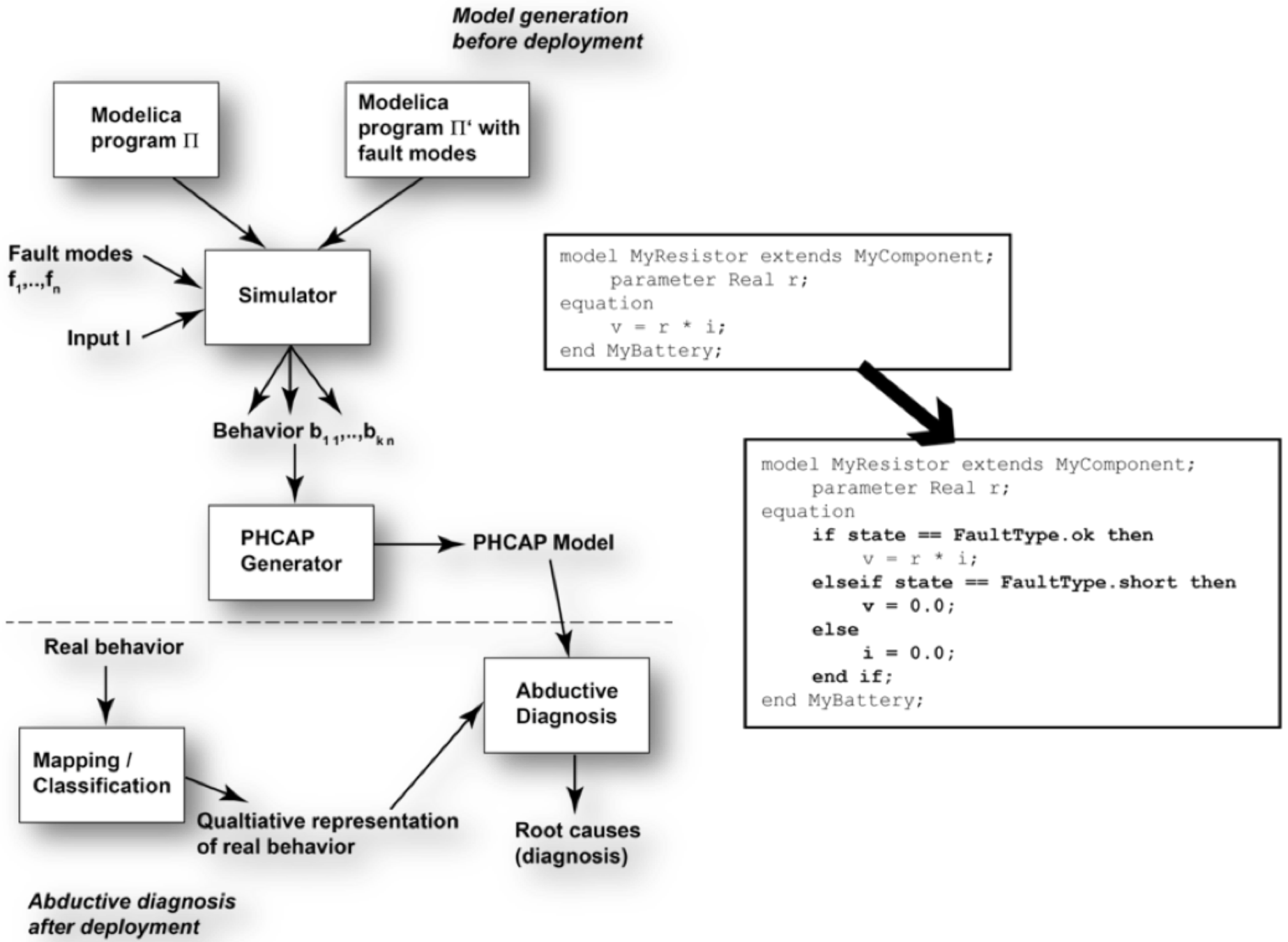
And so on ... finally leading to 2 possible diagnoses statement 3 and statement 4

Automated conversion of programs into constraints

- Assignments are mapped to equations (= constraints)
- Loops are unrolled (= nested if-then-else statements) like in bounded model-checking
- Use the static single assignment (SSA) form to prevent from multiple definitions of variables
- If-then-else statements can be mapped to implications (ako “phi functions” in SSA forms)

❸ Testing

- Bunch of work on testing
  - Security testing (AI planning for test suite generation)
  - Combinatorial testing (e.g. for model extraction from simulation models)
  - Model-based testing (LTS, finite automata & model-checking, STS)
  - White-box testing
  - Test oracles (using LTL and metamorphic testing)
  - Conformance testing of protocols (TLS handshake, VoIP using Lotos and LNT)


Model generation
before deployment
Modelica
program Π
Modelica
program Π‘ with
fault modes
Fault modes
f1,..,fn
Input I
Simulator
Behavior b11,..,bkn
PHCAP
Generator
PHCAP Model
model MyResistor extends MyComponent;
parameter Real r;
equation
v = r * i;
end MyBattery;
model MyResistor extends MyComponent;
parameter Real r;
equation
if state == FaultType.ok then
v = r * i;
elseif state == FaultType.short then
v = 0.0;
else
i = 0.0;
end if;
end MyBattery;
Real behavior
Mapping /
Classification
Qualtiative representation
of real behavior
Abductive
Diagnosis
Root causes
(diagnosis)
Abductive diagnosis
after deployment

Challenges of V&V

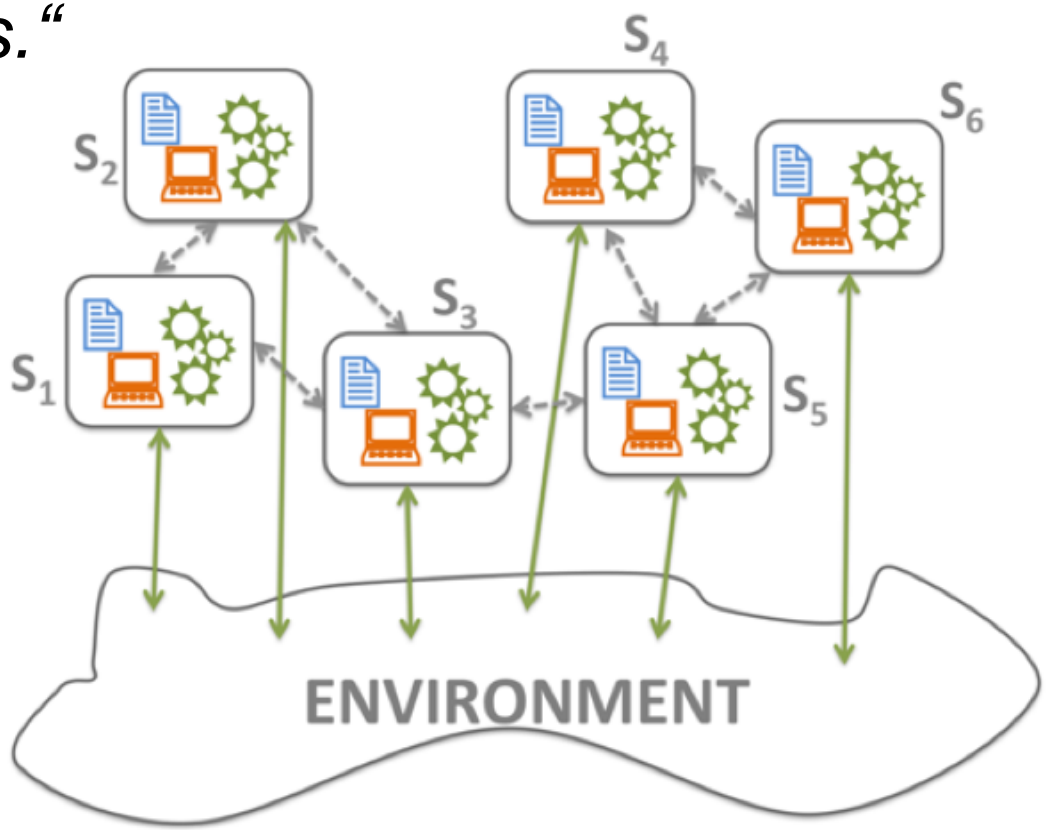
- AI software!
- Autonomous vehicles
- Autonomous robots
- ...

- How to ensure that the software works as intended?
  - Is the software correct? Does it fulfill the requirements and the spec?
  - Is there no unwanted behavior included? (Trojans,…)

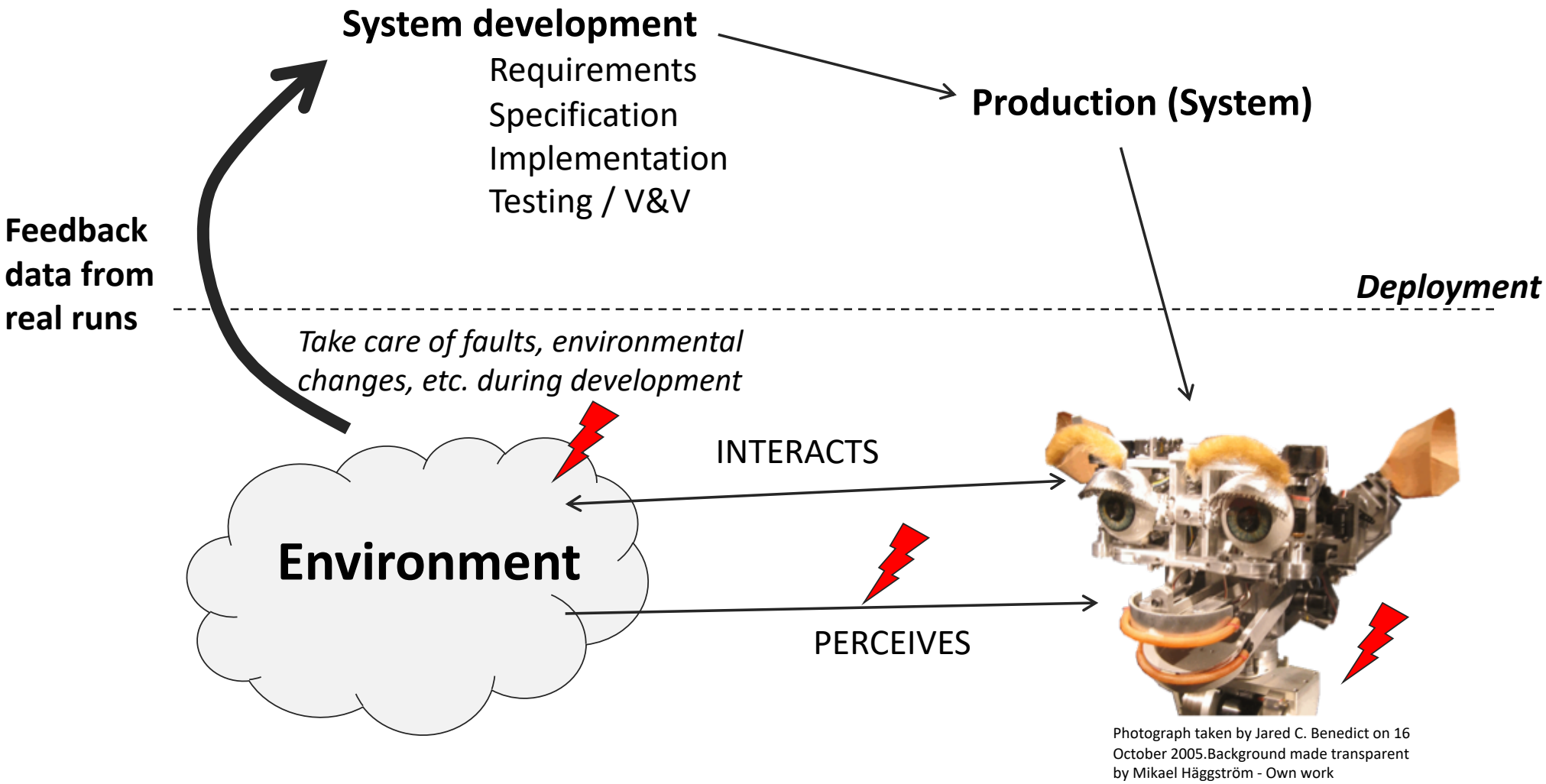
Current project

QAMCAS – Objectives, Goals, Methods

„The vision of QAMCAS is to develop rigorous and evidence-driven .... methodologies to master definition, assessment and prediction of the quality of CPSs.“



Challenges / Vision

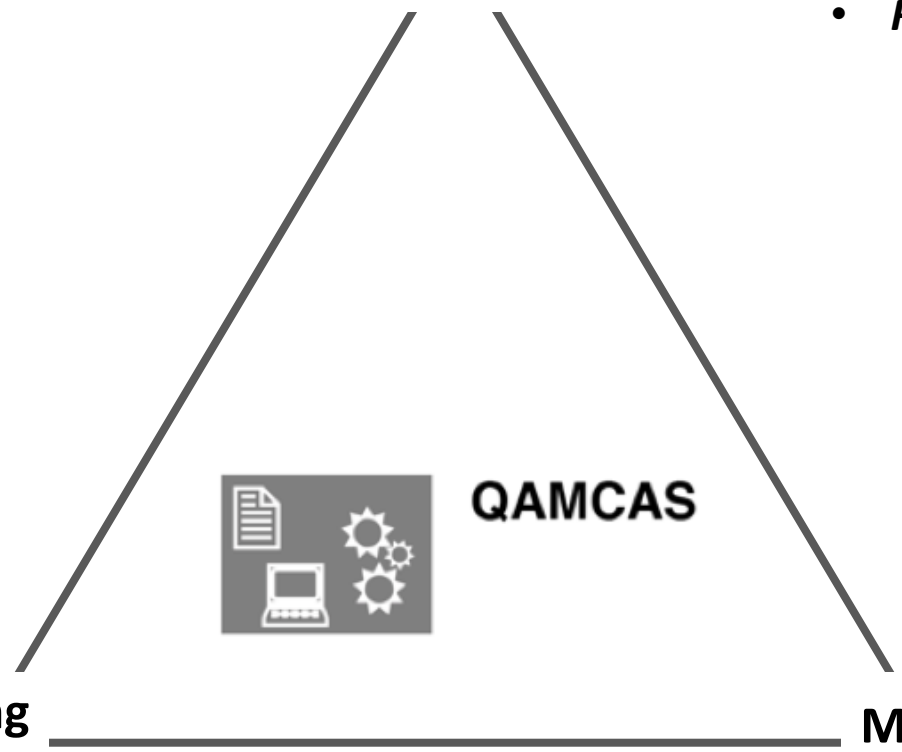


Photograph taken by Jared C. Benedict on 16 October 2005.Background made transparent by Mikael Häggström - Own work

Methods used

Combinatorial Testing (CT)

- *Algorithm improvement (variable strength)*
- *Introducing time*
- *Oracle problem*
- ***Reliability measure***



- ***Anytime algorithms***
- ***Use models from engineering***
- *Consequences of model faults*
- ***V&V for self-* systems***

Model-based Reasoning (MBR)

- ***Indicator for test suite quality***
- *Behavioral similarity*
- *Find appropriate learning methodology*
- *Test suite improvement*

Machine Learning (ML)

What is important?

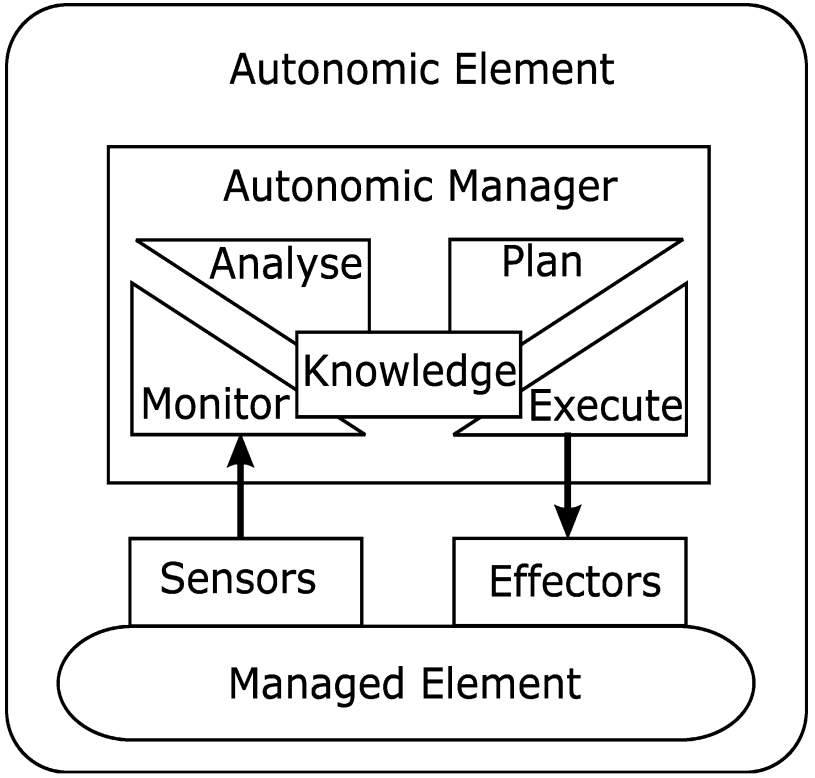
- Quality assurance
  - During development / design
  - During operation

- E.g. we have to guarantee that a test suite is good enough!
  - Source code coverage
  - Mutation score
  - Combinatorial strength
  - Formal verification
  - Learning-based measures (very recent)

What is important?

- Self-* capabilities
- Fail-operational (and still fail-safe)

- E.g. how to guarantee fail-safe behavior for the *MAPE-K (Monitor, Analyse, Plan, Execute, Knowledge)* architecture*?*



What are we interested in?

- **Dependable Systems**:
  - Increased reliability (design faults vs. faults caused from aging)
    - Correctness via testing and other quality assurance methodologies
  - Increased availability via self-* capabilities
  - Safety & security

- Techniques & methods to be applied during design/development and runtime! (and how to bring them together like in runtime verification)


Combining AI and SE for building safer and more reliable systems
Questions?