Integration of Runtime Verification into Metamodeling

F. Macias   T. Scheffel   M. Schmitz   **R. Wang**
M. Leucker   A. Rutle   V. Stolz

28th Nordic Workshop on Programming Theory (NWPT'16), Denmark

Why Runtime Verification?

- DSML do not shield the software from design errors
- Runtime Verification checks the execution of real system
  - Consider environmental influences
  - React to failures
- Testing is seldom exhaustive
- Model Checking can not always gurantee the correctness of executing system

Runtime Verification

- Use runtime monitors to observe the run of systems
- Check whether the current execution of such systems violates given correctness properties
- Such correctness properties can be formulated in linear-time temporal logic, LTL

Integrating RV into Modeling

Goal

⇒ Integrate RV into software engineering process
⇒ Domain experts and verification engineers use same model
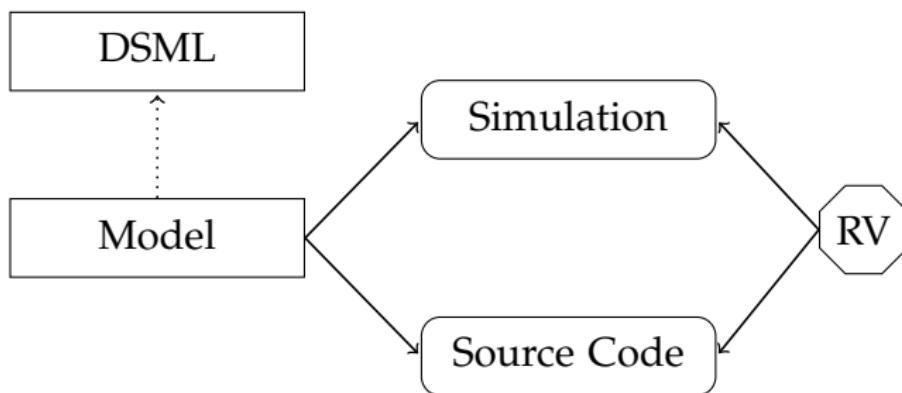
Multilevel Metamodeling

Example Scenario

$$\mathcal{G}(\text{obs} \rightarrow \mathcal{X}(\neg \text{ obs } \mathcal{U} \text{ to}))$$

Obstacle
GoBck — TurnL
Start
GoFwd — Timeout
Border
GoBck — TurnR
Input
Task
Transition
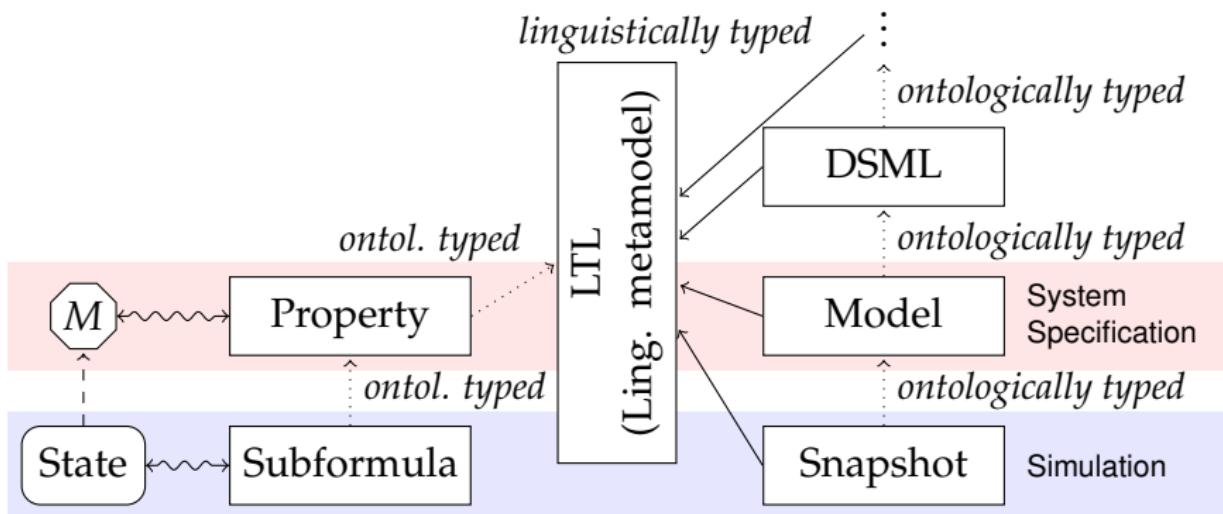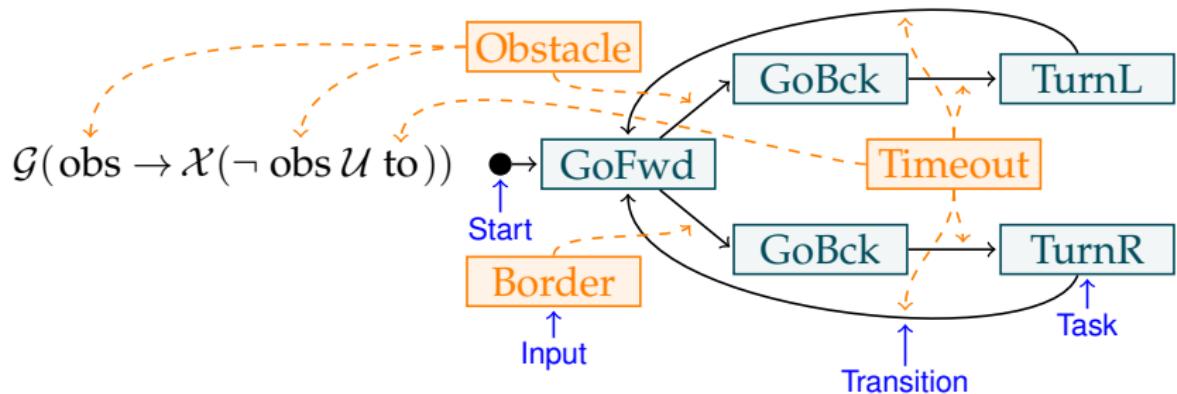
▶ Generate Python code for the robot and monitor

Example Execution
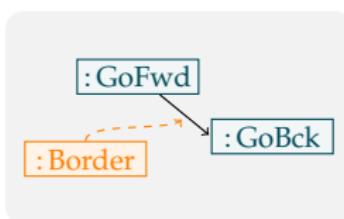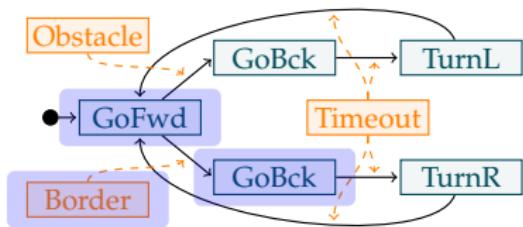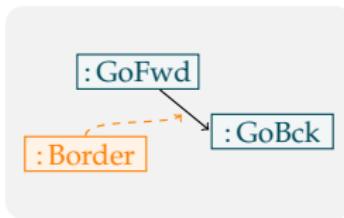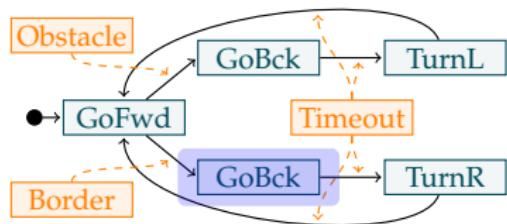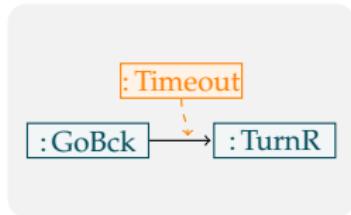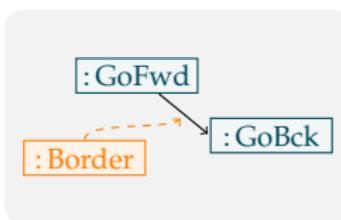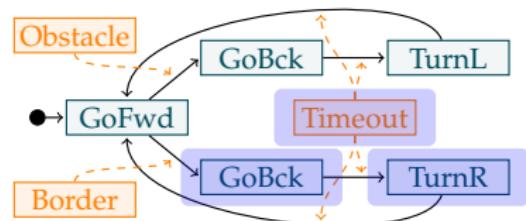
Example Execution

Example Execution

Example Execution
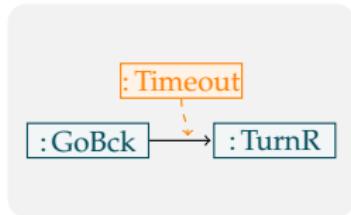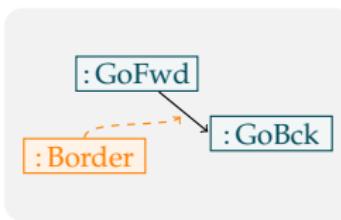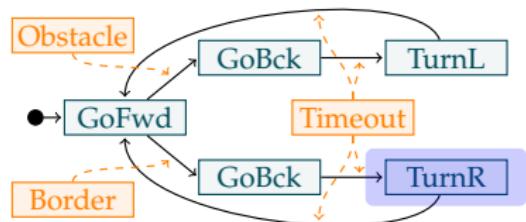
Example Execution
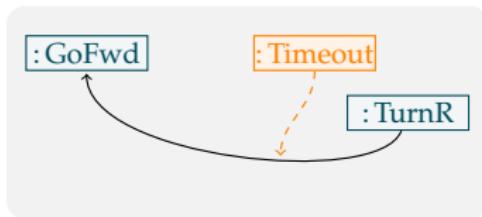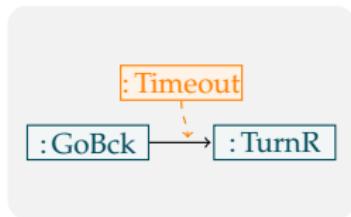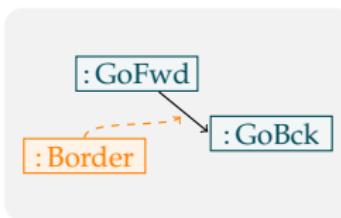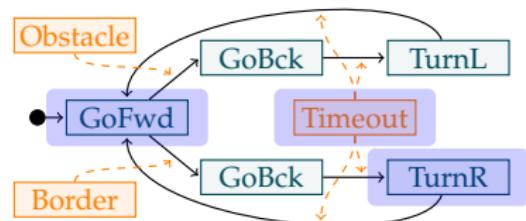
Example Execution
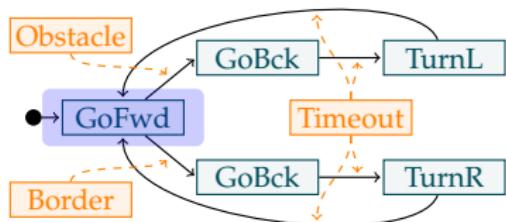
Example Execution II

$$\mathcal{G}(\text{obs} \rightarrow \mathcal{X}(\neg \text{obs}\,\mathcal{U}\,\text{to}))$$

Example Execution II
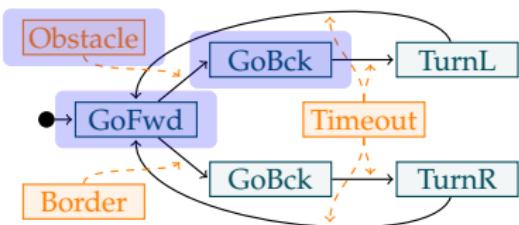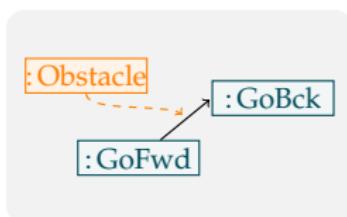
$\mathcal{G}(\text{obs} \to \mathcal{X}(\neg\,\text{obs}\,\mathcal{U}\,\text{to}))$

$\mathcal{G}(\text{obs} \to \mathcal{X}(\neg\,\text{obs}\,\mathcal{U}\,\text{to}))$

Example Execution II
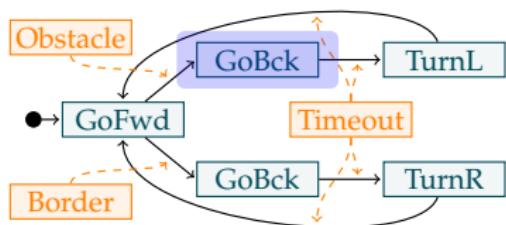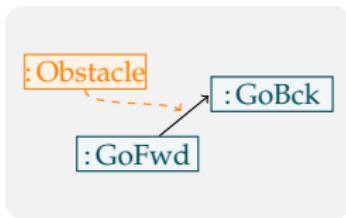
$\mathcal{G}(\text{obs} \to \mathcal{X}(\neg \text{obs} \, \mathcal{U} \, \text{to}))$

$\mathcal{G}(\text{obs} \to \mathcal{X}(\neg \text{obs} \, \mathcal{U} \, \text{to}))$

$\neg \text{obs} \, \mathcal{U} \, \text{to}$

Example Execution II
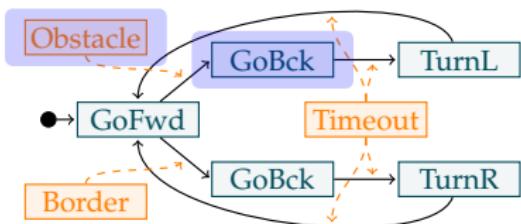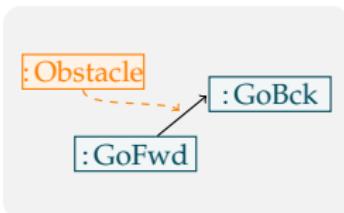

$\mathcal{G}(\text{obs} \rightarrow \mathcal{X}(\neg\,\text{obs}\,\mathcal{U}\,\text{to}))$



$\mathcal{G}(\text{obs} \rightarrow \mathcal{X}(\neg\,\text{obs}\,\mathcal{U}\,\text{to}))$


$\neg\,\text{obs}\,\mathcal{U}\,\text{to}$

$\neg\,\text{obs}\,\mathcal{U}\,\text{to}$

Coupled Model Transformation Rules
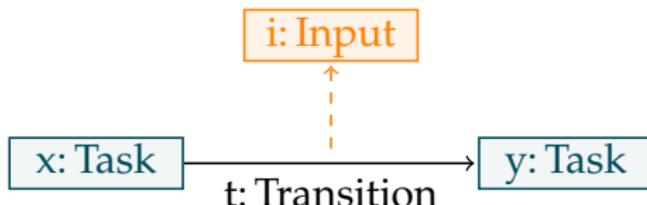
META

FROM	TO

Model Transformation Rules
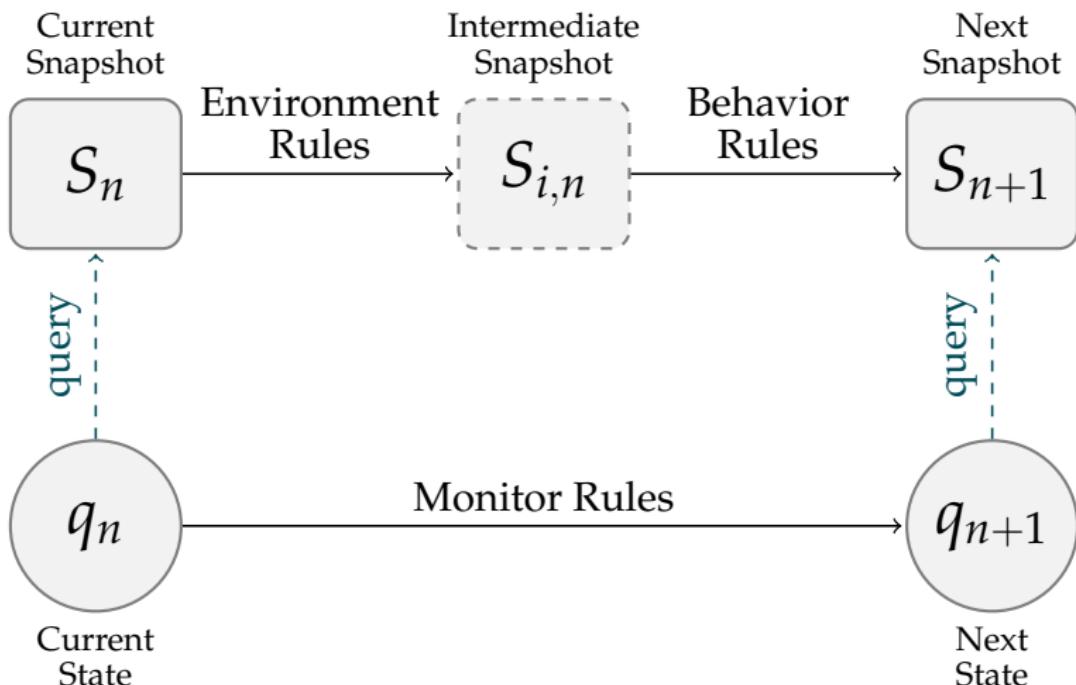
Example Video
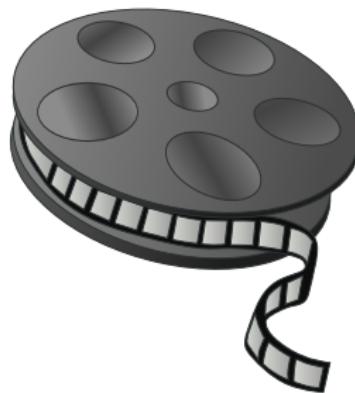
Communication

Goal

⇒ Model distributed systems

- ▶ Current scenario: two robots meet
- ▶ Client/Server Model
- ▶ Sockets - send and receive data
- ▶ UDP (The User Datagram Protocol)
  - ▶ No handshaking dialogues
  - ▶ No guarantee of delivery, ordering, or duplicate protection
  - ▶ Suitable for purposes where error checking and correction are performed in the application
  - ▶ Broadcasting information

Broadcast

- Romba/Spinner simple "Found" message
- More advanced messages for robots explore area and share updates to a common map
- Broadcast Hello messages (ID, timestamp and data)
- The number of retransmission, reTx, is counted
- Maximum number of transmissions: MaxReTx

Example Scenario for communication
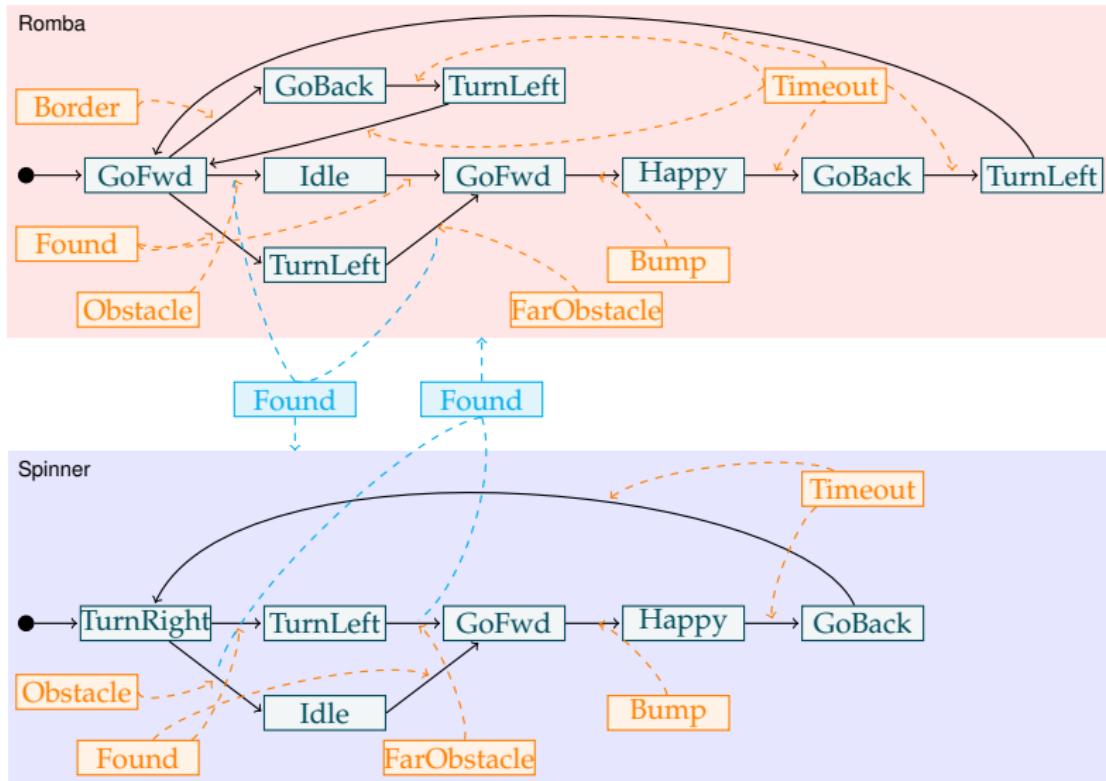
Conclusion and outlook

- ► Present a metamodel that captures
  a wider range of aspects of the robots
  - ► Sensors
  - ► Motors
  - ► Communication
- ► Integrate runtime verification into the whole
  software engineering process
  - ► Design
  - ► Simulation
  - ► Code generation for the robot and monitor
- ► Allow the design of distributed systems in the future
  - ► Distribution can be modeled through replication
    of existing instances on the modeling level
- ► Use models for Model-based testing