Queen's School of Computing
Queen's University
Kingston, Ontario, Canada

Queen's University

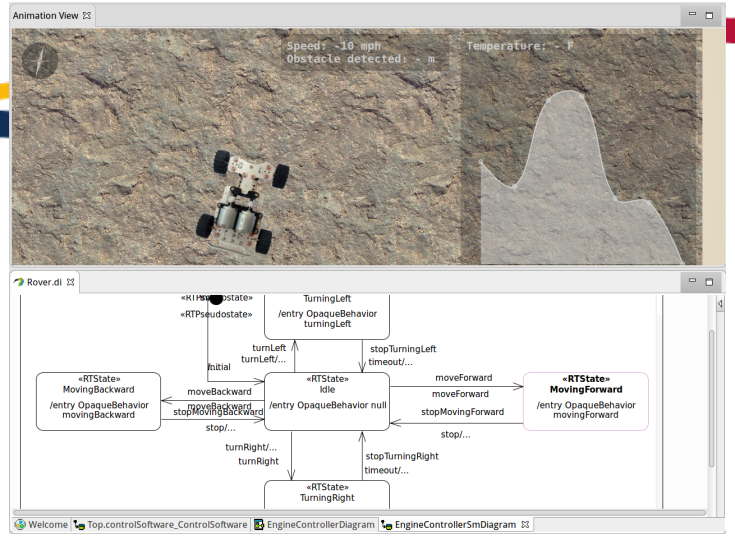
I am therefore I compute

Model-Level Support for Observation and Steering

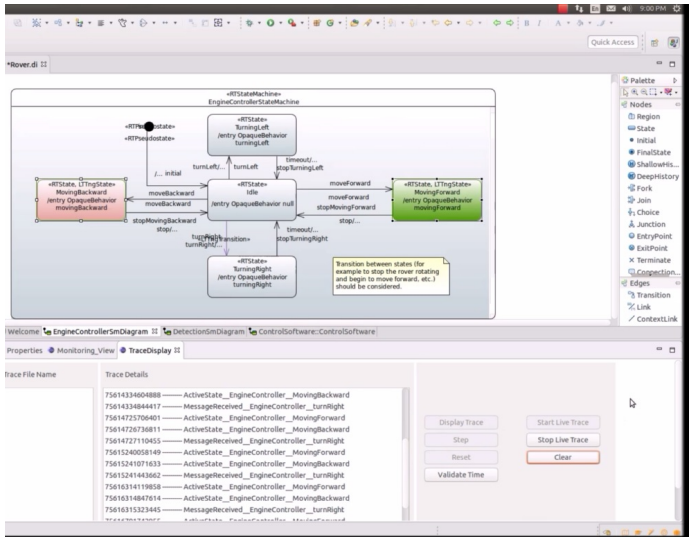
Nicolas Hili, Mojtaba Bagherzadeh, Juergen Dingel
{hili, mojtaba, dingel}@cs.queensu.ca
School of Computing, Queen's University,
Kingston, Ontario, Canada

Progress Report Meeting, Montreal
December, 2016

Overview of the Current Development

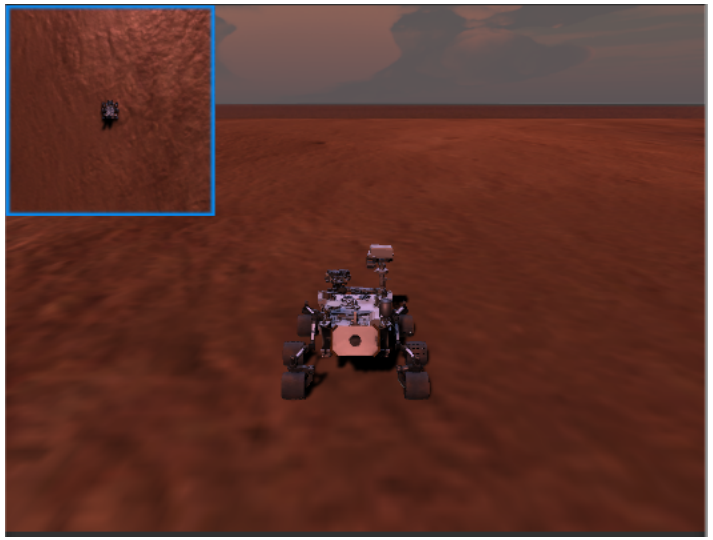


Model Animation in a 2D environment (Web-based)



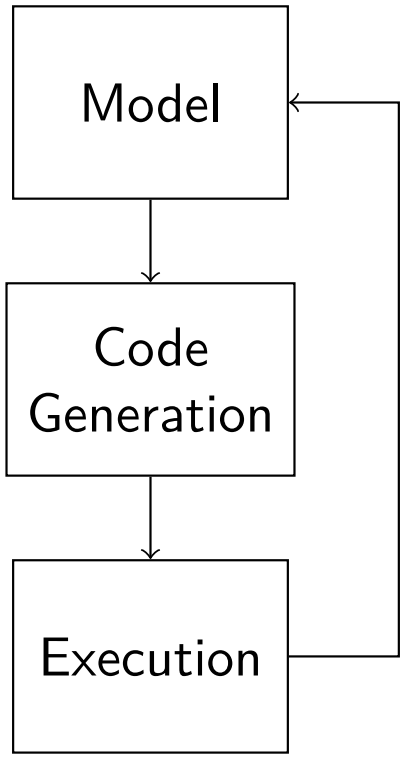
Model Monitoring of timing constraints (LTTng)

Common denominator ?



Model Animation in a 3D environment (Unity)

Common Denominator (1): Model-based Approach



Models as first-class artifacts...

✓ System is modeled in UML-RT:

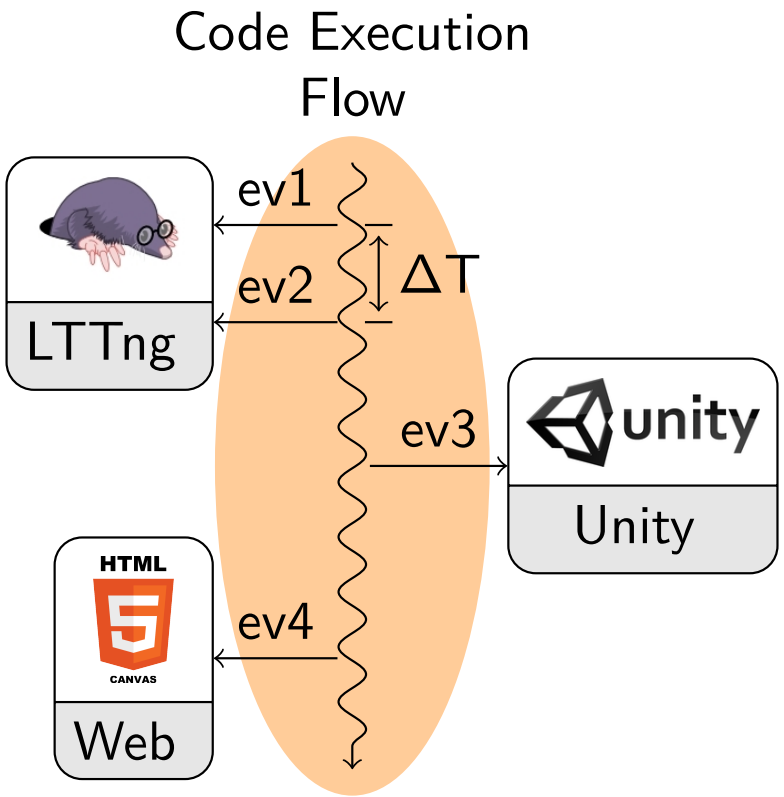
- Code of the system is fully generated
- Don't expose the complexity of the code

▼

...Allowing for monitoring and observation

☞ Monitoring different aspects of the system
☞ Animating the system execution for a better understanding
☞ Software-in-the-loop
☞ etc.

Common Denominator (2): Support for Monitoring



Not only for observing the system execution...

✓ Each tool acts as an *observer*:

- Listens for specific events
- Does not disrupt the execution flow
- Consumer / Producer paradigm

▼

...but also for steering it !

✓ Feed the system execution by injecting events from external tools.

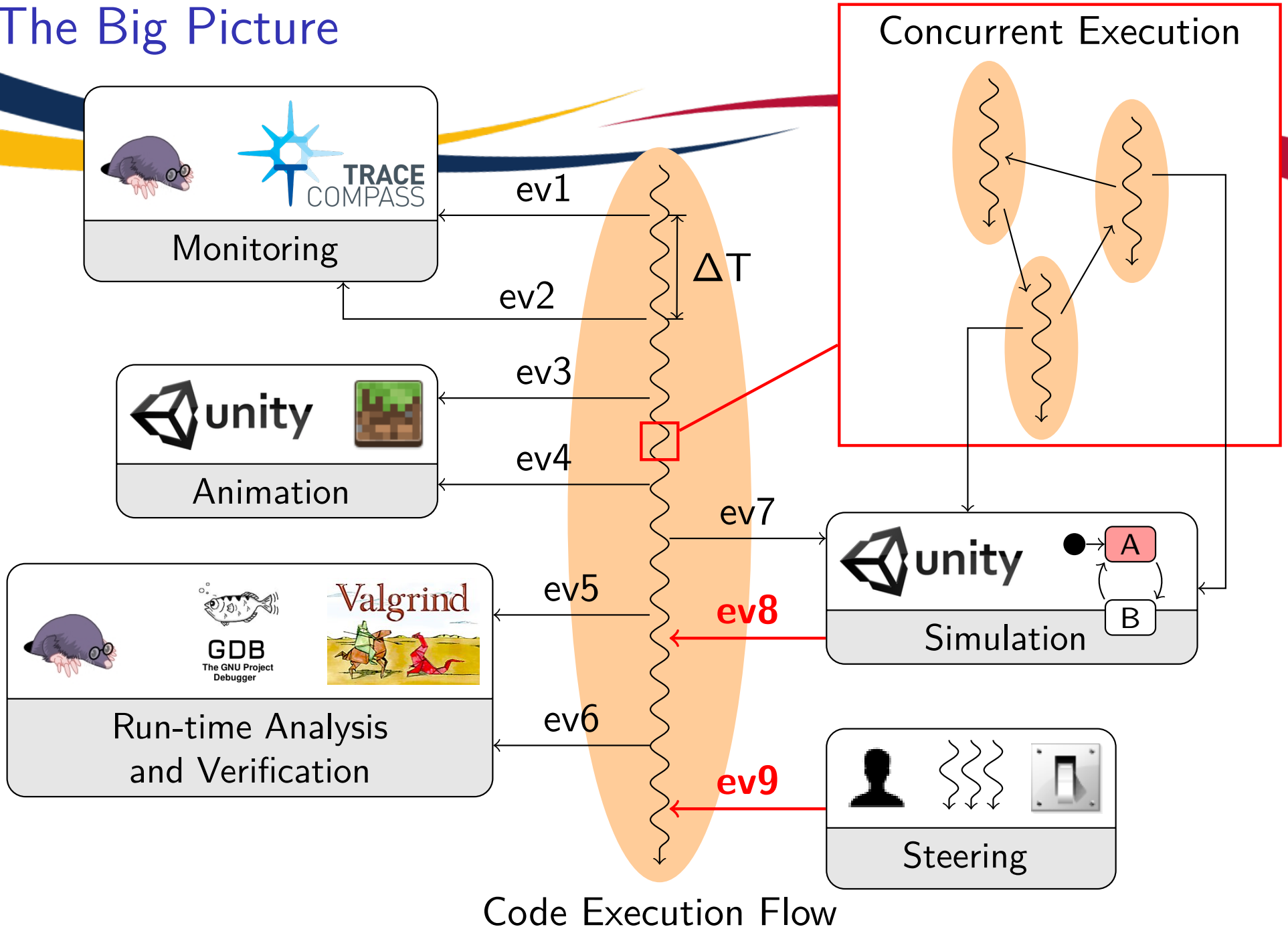
☞ Need for a model-level support for **observation** and **steering**.

Contribution:

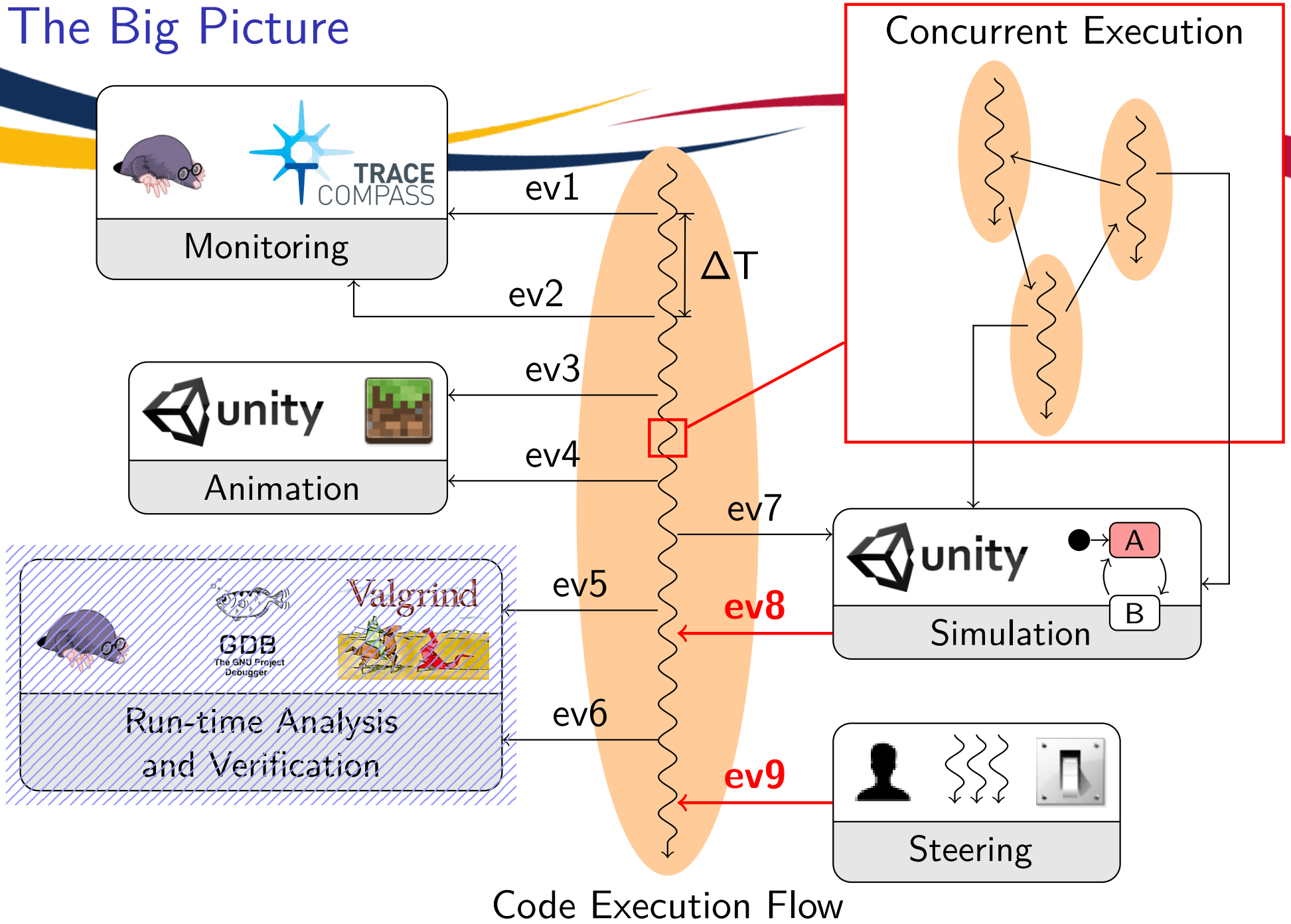
Model-based architecture for supporting observation and steering of UML-RT Models

and its integration into Papyrus-RT

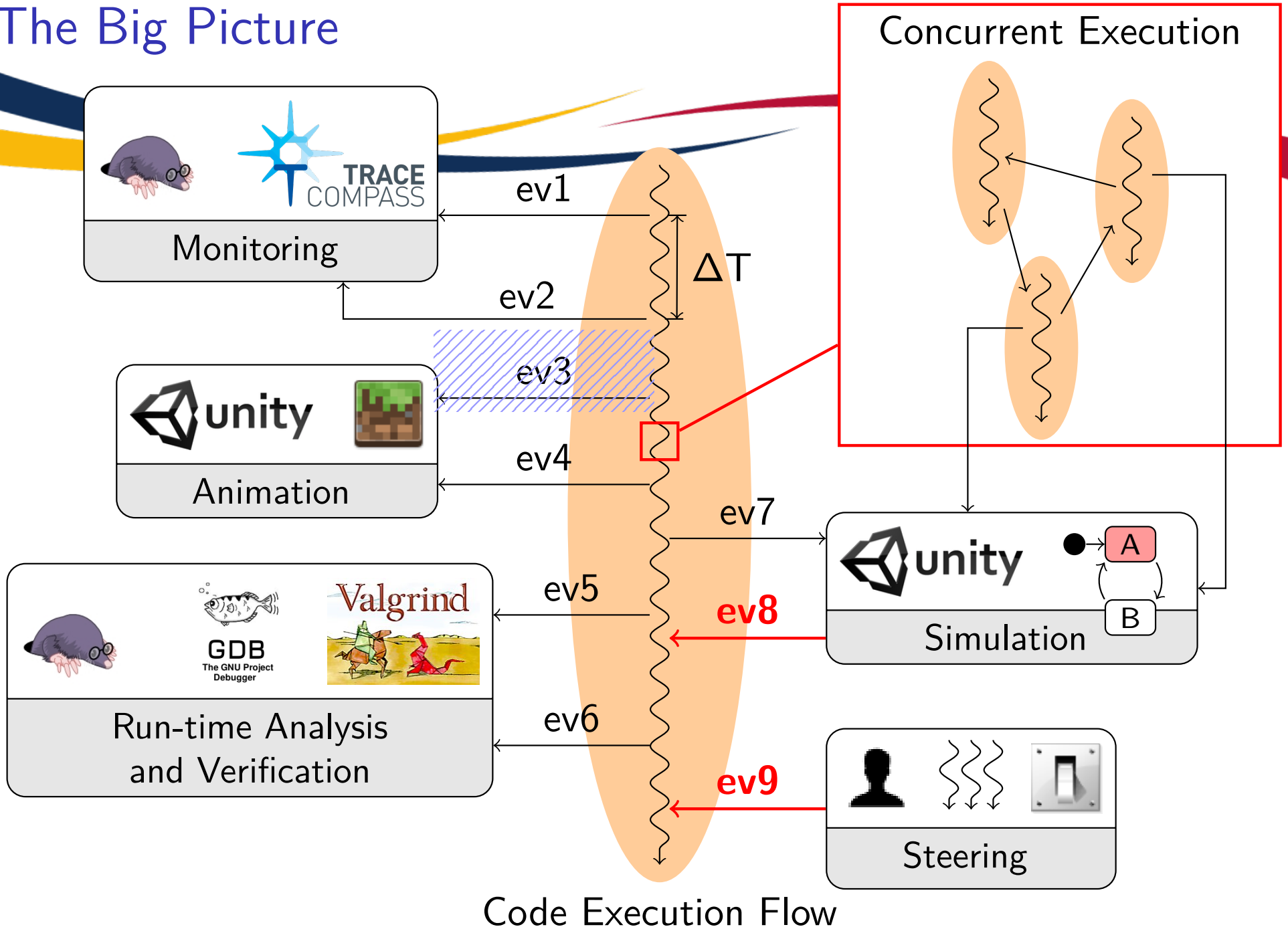
The Big Picture



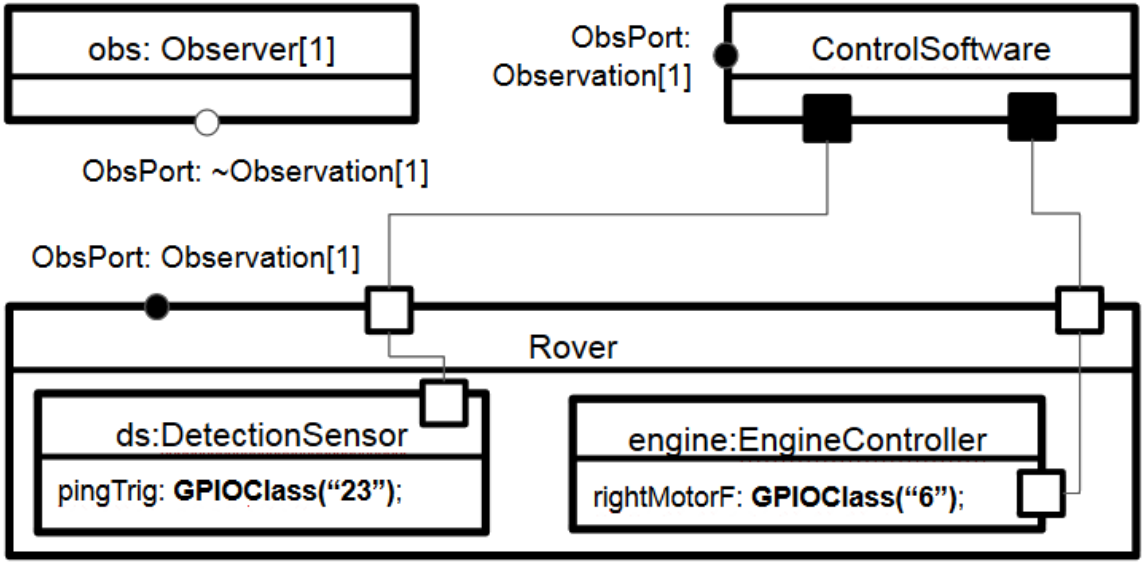
The Big Picture



The Big Picture



Implementation: Observer Capsule in UML-RT



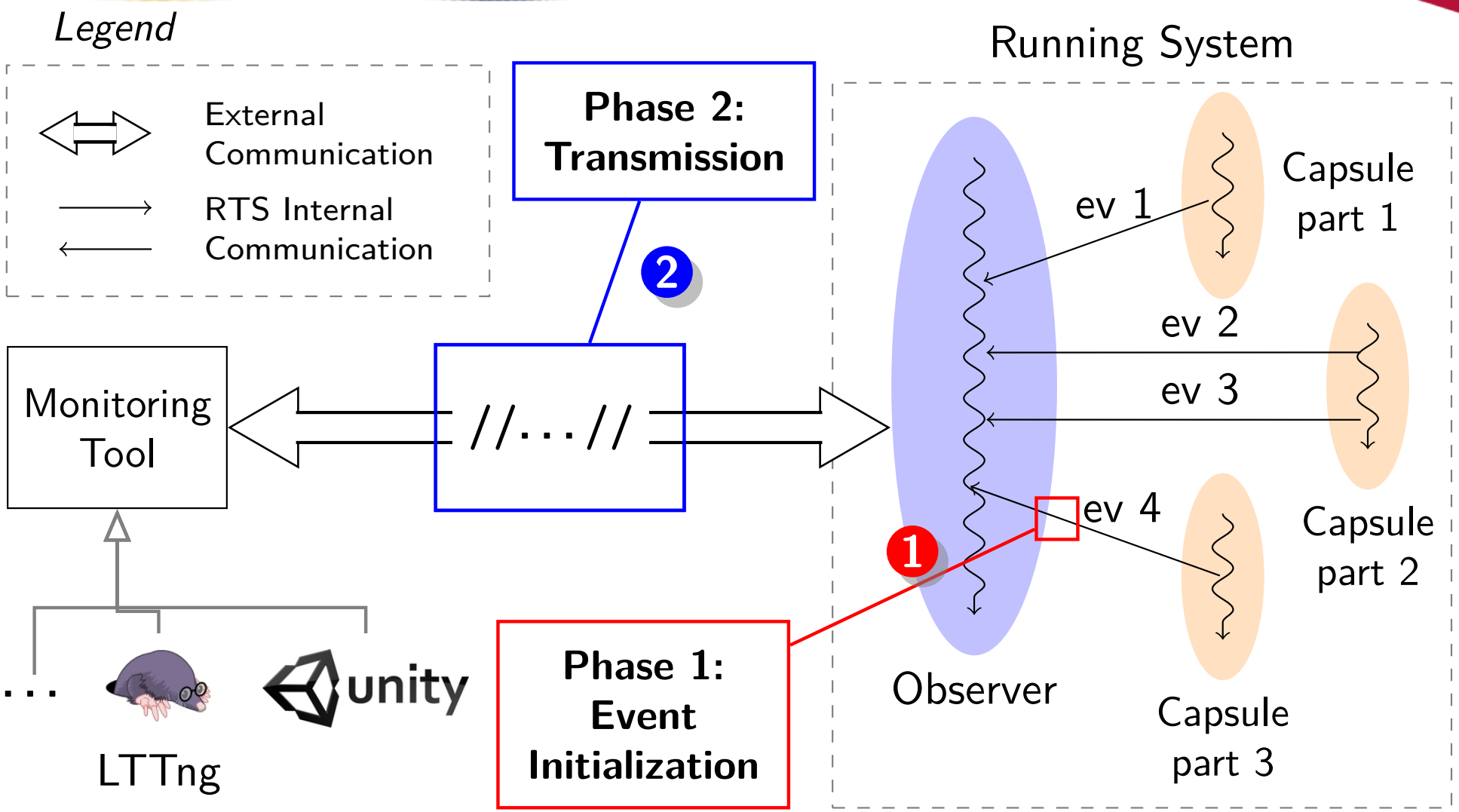
What is it ?

✓ Capsule implemented in UML-RT;
✓ Rely on the SAP / SPP communication;
✓ Intended to be integrated into the Papyrus-RT RunTime Service.

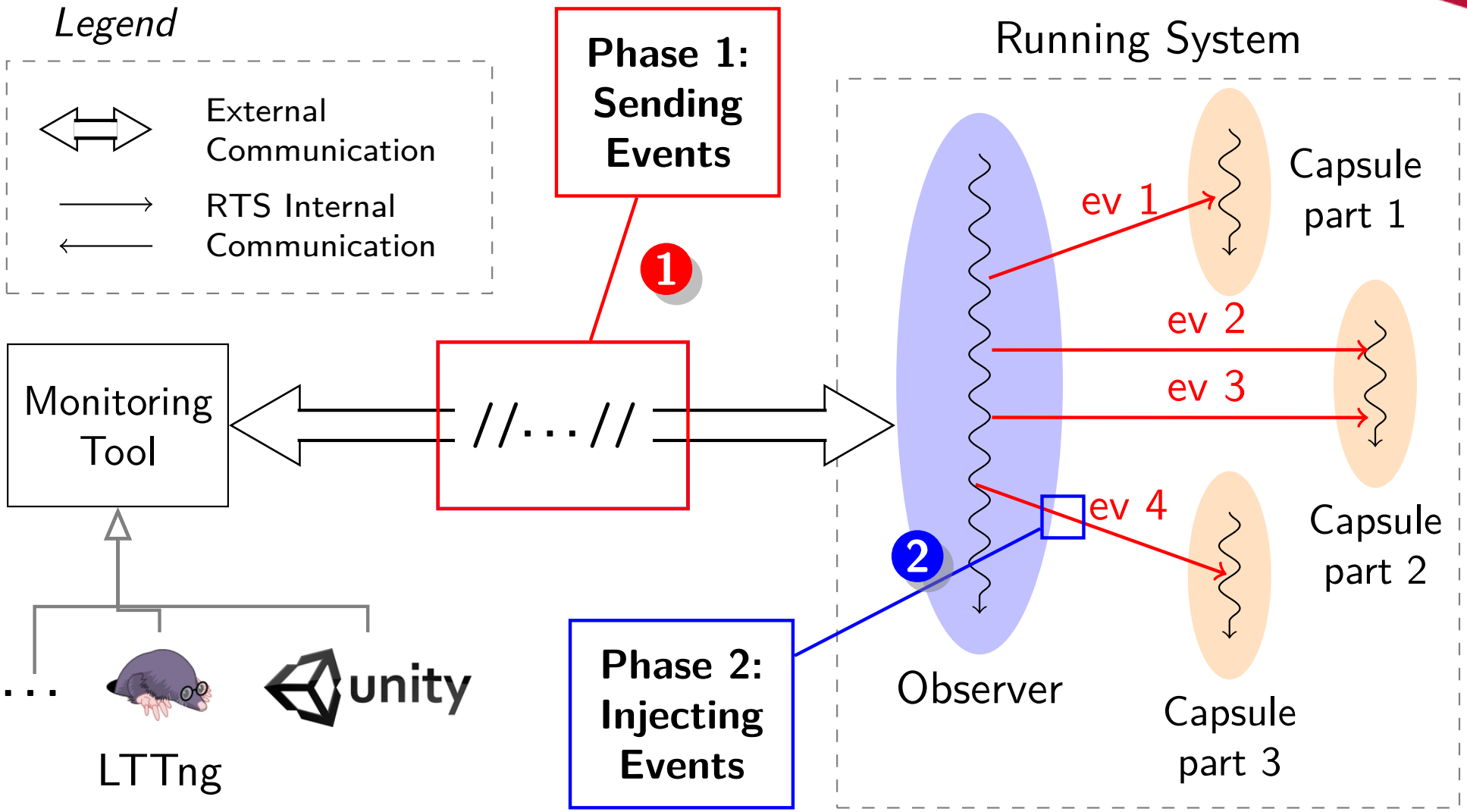
What for ?

✓ Gathering events from registered capsules;
✓ Transmitting events to external monitoring tools in a unified way;
✓ Reversible: events can be injected into the running system.

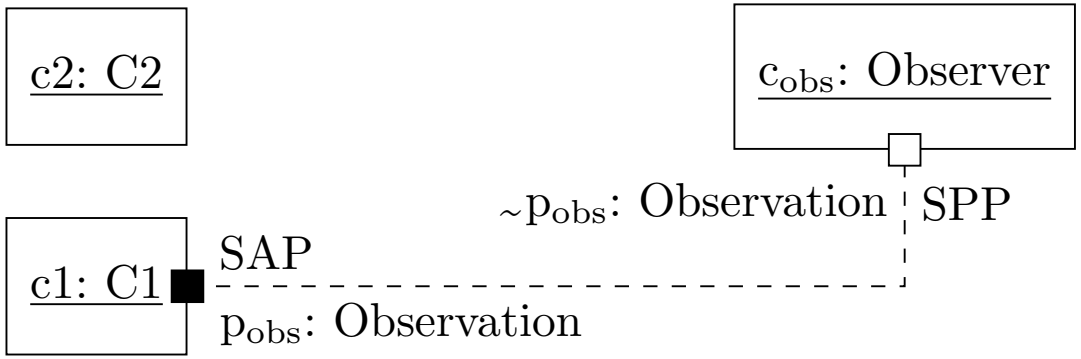
How it works: Observing



How it works: **Steering**

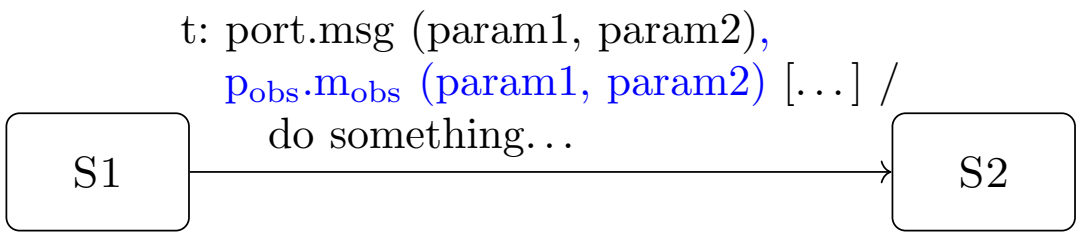


How to Use: Preparing UML-RT Models



Observing:

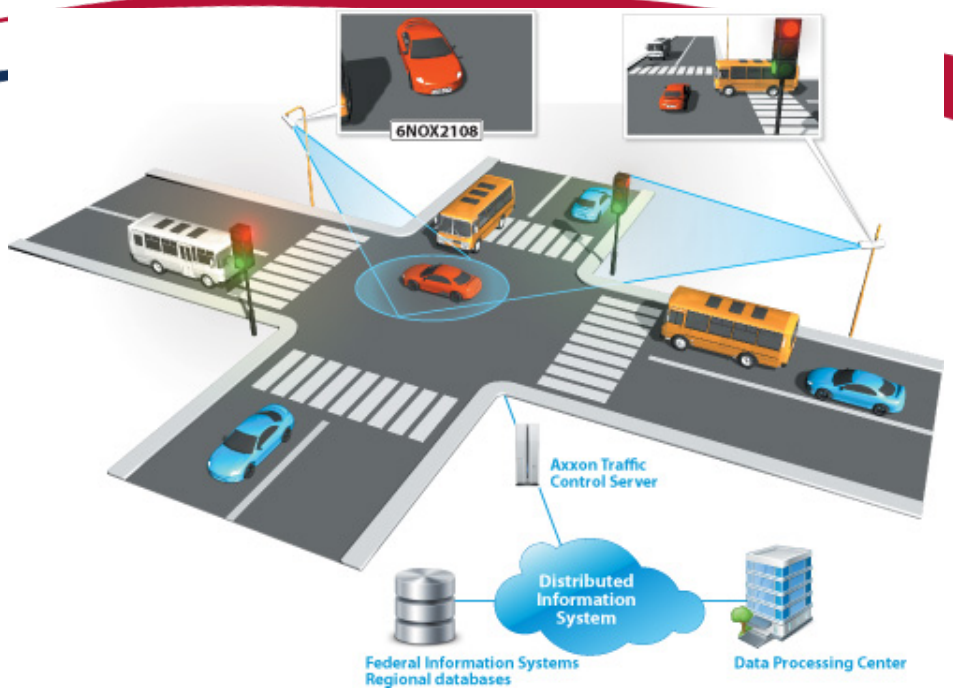
- Observer capsule;
- SAP / SPP ports.



Steering:

- Trigger duplication;
- Commands sent by the Observer capsule through this trigger.

Infrastructure's Challenges



- Distributed system monitoring
- Taming time
- Security

Monitoring distributed systems is a challenging problem.

☞ Different execution mechanisms
☞ Different clocks
☞ ...

Infrastructure's Challenges

- Distributed system monitoring
- Taming time
- Security

Sometimes the execution is too long to be monitored.

☞ Alter the base time

☞ Alter / Pause the execution

☞ ...

Infrastructure's Challenges

- Distributed system monitoring
- Taming time
- Security

May open backdoors for intruders !

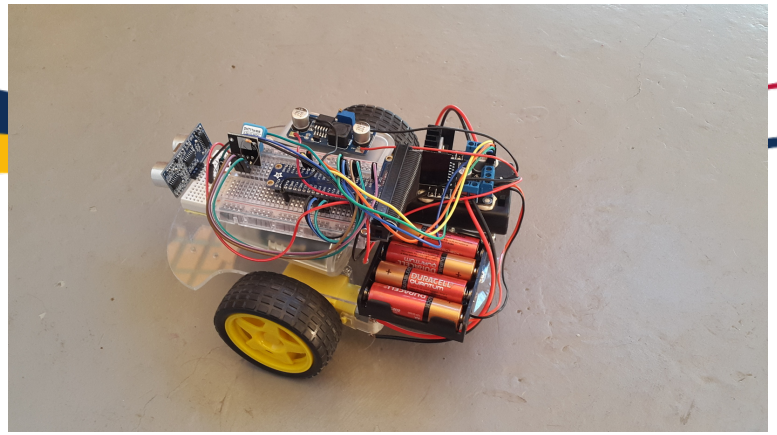
☞ Authorizing trustworthy tools only

☞ Preventing malicious software from altering the execution flow

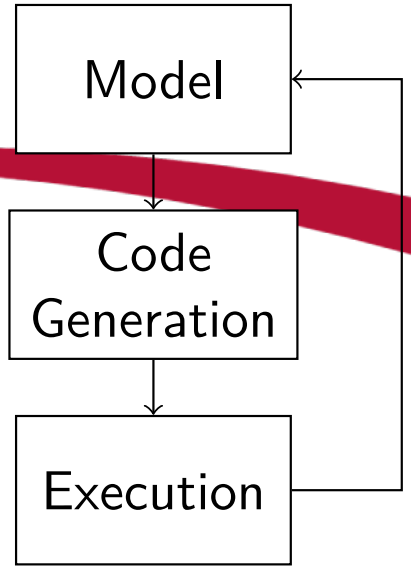
☞ ...

Conclusion

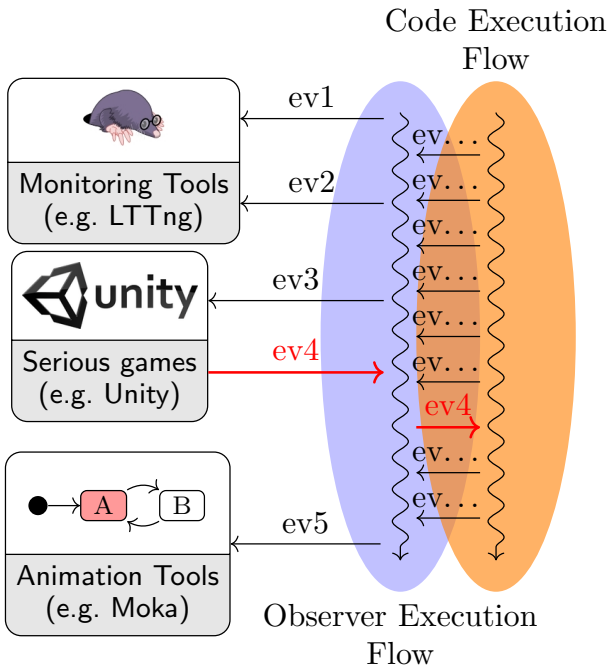
Summary of the Contributions

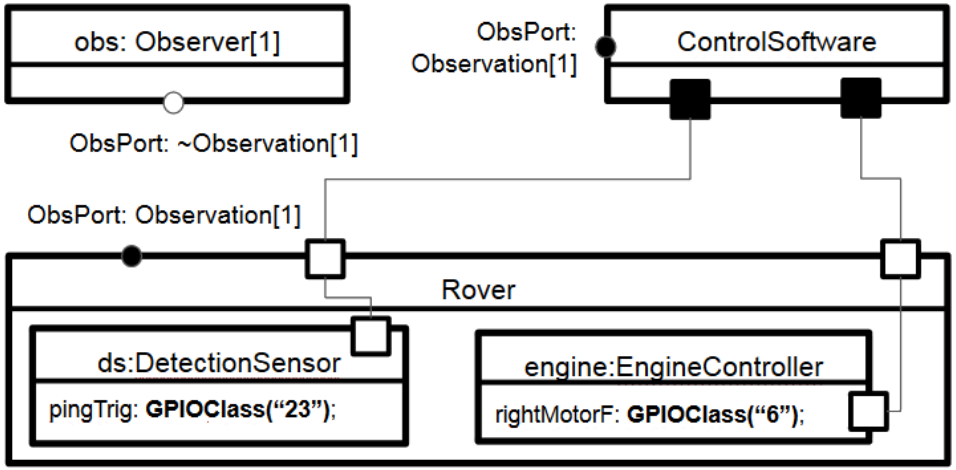


Rover Case Study *(Ahmadi et al., OSS4MDE'16)*



Vision paper *(Das et al., Models'16)*





Observer Architecture *(Hili et al., Draft paper)*

Use cases and Ongoing Work

Different use cases:

- *Run-time monitoring with LTTng:* `https://youtu.be/hi5wFC7X4h8`
- *Run-time animation in a web-based environment:* `https://youtu.be/OMQxG33q3PQ`
- *Run-time animation in 3D environments with Unity (Michal)*

One demonstrator:

- *The Rover System:* `https://youtu.be/hi5wFC7X4h8`

Publications:

- Two accepted papers, one in the pipe, more to come

Ongoing work:

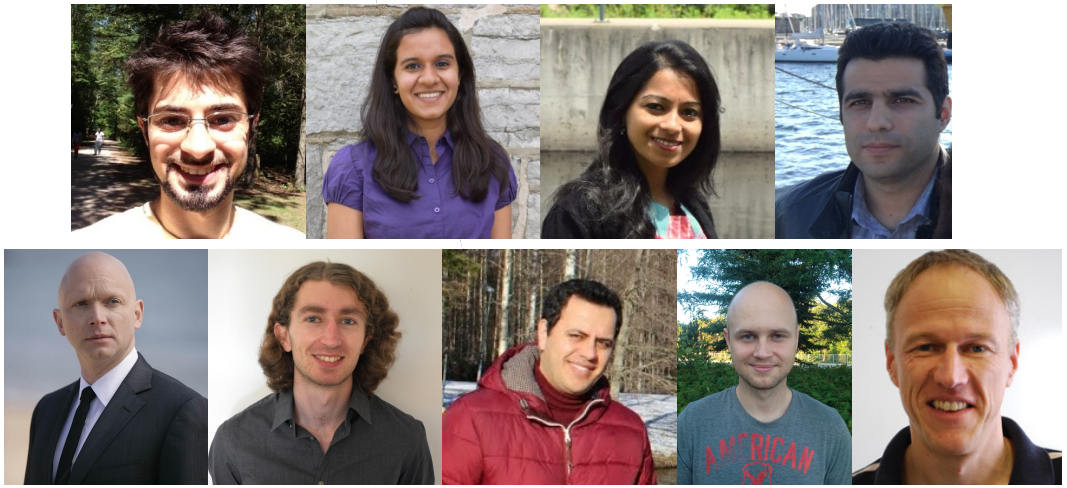
- Model-based Debugging (Mojtaba)
- Run-Time Verification (Reza)

Queen's School of Computing
Queen's University
Kingston, Ontario, Canada

Queen's University

Sum Ergo Computo
I am therefore I compute

Thank you !



Real observer

Questions ?