Traceability Support in OpenModelica Using Open Services for Lifecycle Collaboration (OSLC)

Alachew Mengist, Adrian Pop,
Adeel Asghar, Peter Fritzson

MODPROD 2017, Linköping

Agenda

- Problem Statement
- Motivation and Approach
- Open Services for Lifecycle Collaboration (OSLC)
- Traceability Design and Architecture
- Traceability and Model Management Workflow in OpenModelica
- Prototype Implementation
- Conclusions and Future Work

Problem Statement

- Large scale system modeling
  - Heterogeneous models (e.g. requirements models, architectural models)
  - Special purpose modeling tool
- Weak or no integration between tools
- Minimal or no traceability between artifacts
- Seamless tracing of the requirements and associating them with the models and the simulation results
  - Traced artifacts: SysML blocks, requirements, models, FMUs, connection configuration, simulation results

Motivation and Approach

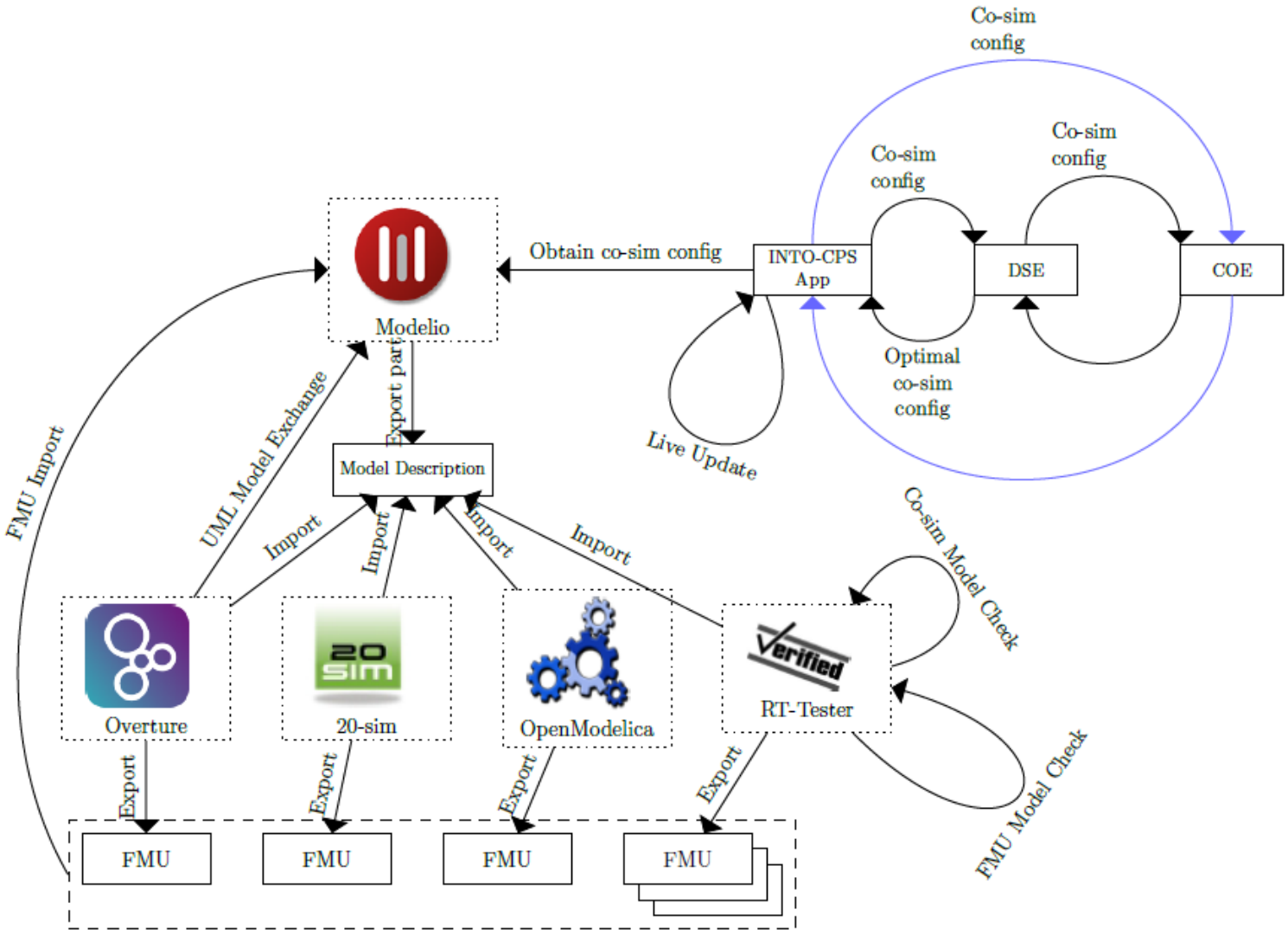
- Why?
  - To trace how high level requirements are validated during system design via simulation,
  - Traceability from requirements to models to simulation executables to simulation results to …
  - Documentation of the system development

- How?
  - All tools are storing information in GIT and sending information about existing and created artifacts to the global database
  - Using OSLC

The INTO-CPS Project and Tooling

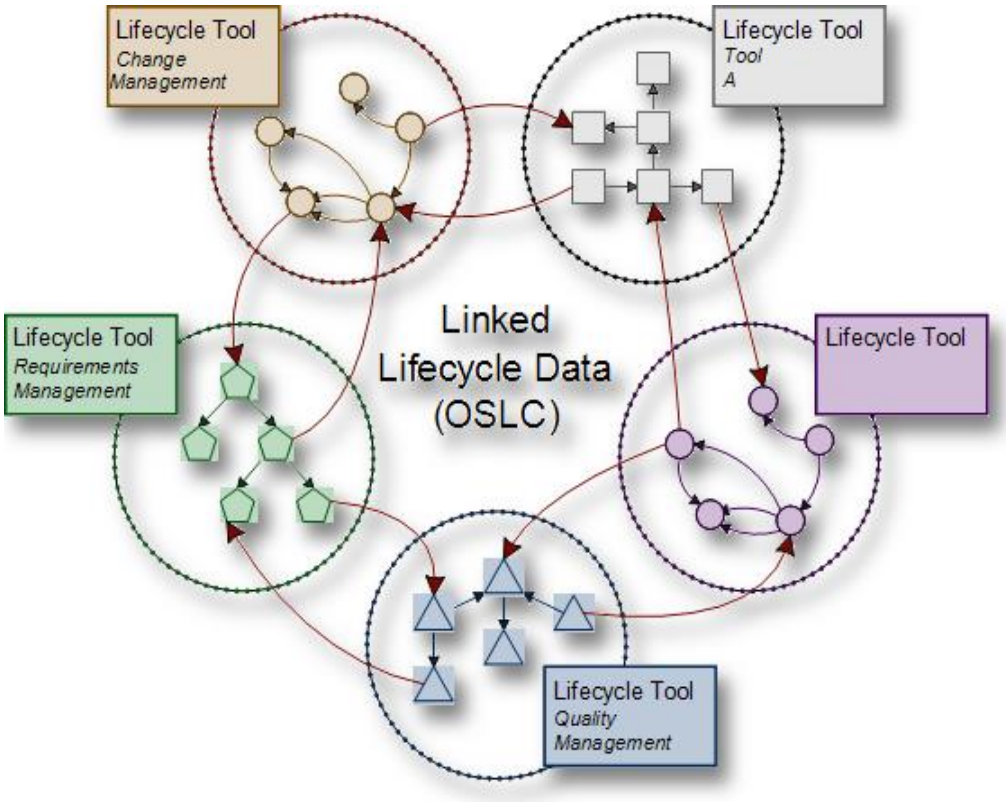
- A project about integrated tools for cyber-physical model development
- Modeling Tools like Overture, 20-sim, OpenModelica are integrated
- High level system architecture developed in SysML, used e.g. to generate Modelica models
- Generated Modelica model is completed with behavior for the SysML block and the final model is exported in the FMU form
- The generated FMU is then used in a whole system simulation connected according to the SysML connection diagram.
- The COE (the FMU master simulation algorithm) component performs the simulation via the INTO-CPS App.

INTO-CPS Tool Chain Connections



Open Services for Lifecycle Collaboration (OSLC)

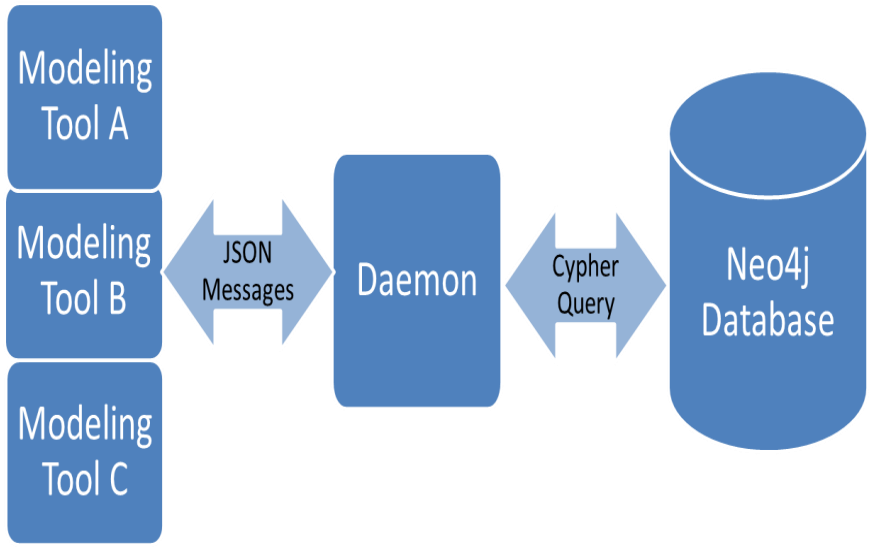
- Enable integration of development life cycle tools
- Based on Linked Data
  - Use URIs as names for things
  - Use HTTP URIs so that people can look up those names
  - When someone looks up a URI, provide useful information using RDF
  - Include links to other URIs. so that they can discover more things
- Create using HTTP POST on creation factory URI
- Query using HTTP GET on query base URI



Source- https://en.wikipedia.org/wiki/Open_Services_for_Lifecycle_Collaboration)

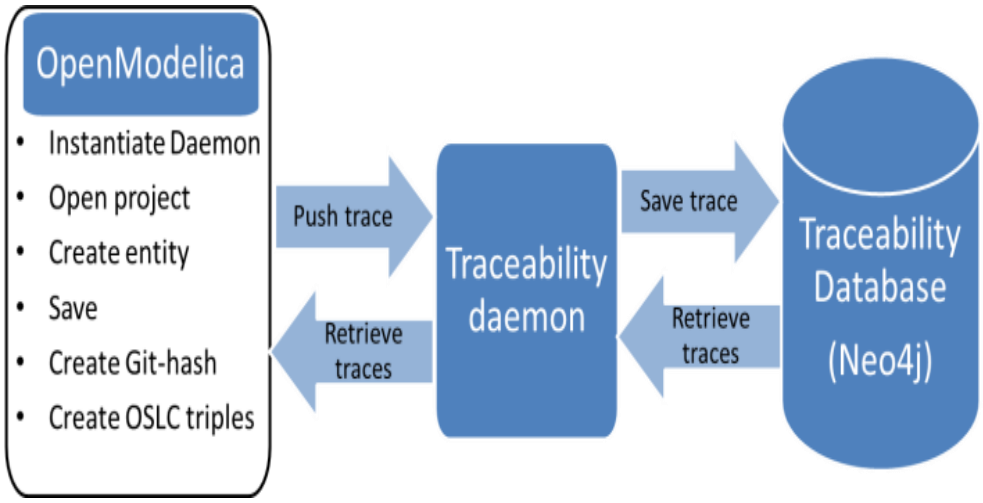
Traceability Design and Architecture

- Modeling Tools
  - Send traceability information to Daemon
  - Query (traces to, traces from) via Daemon
- The Daemon
  - Provides an OSLC interface with RESTful interface
  - OSLC triples(Entities, activities and agents) in JSON format
- Neo4j Database
  - Graph database to store the OSLC triples



Traceability and Model Management Workflow in OpenModelica

- Commit model file entity to Git repository and record the Git-hash
- Create URIs of the activity based on the Git-hash
- OSLC triples describing the activity are generated using the URIs
- OSLC triples are sent to the traceability Daemon
- Retrieve the traceability information (traces to and traces from)

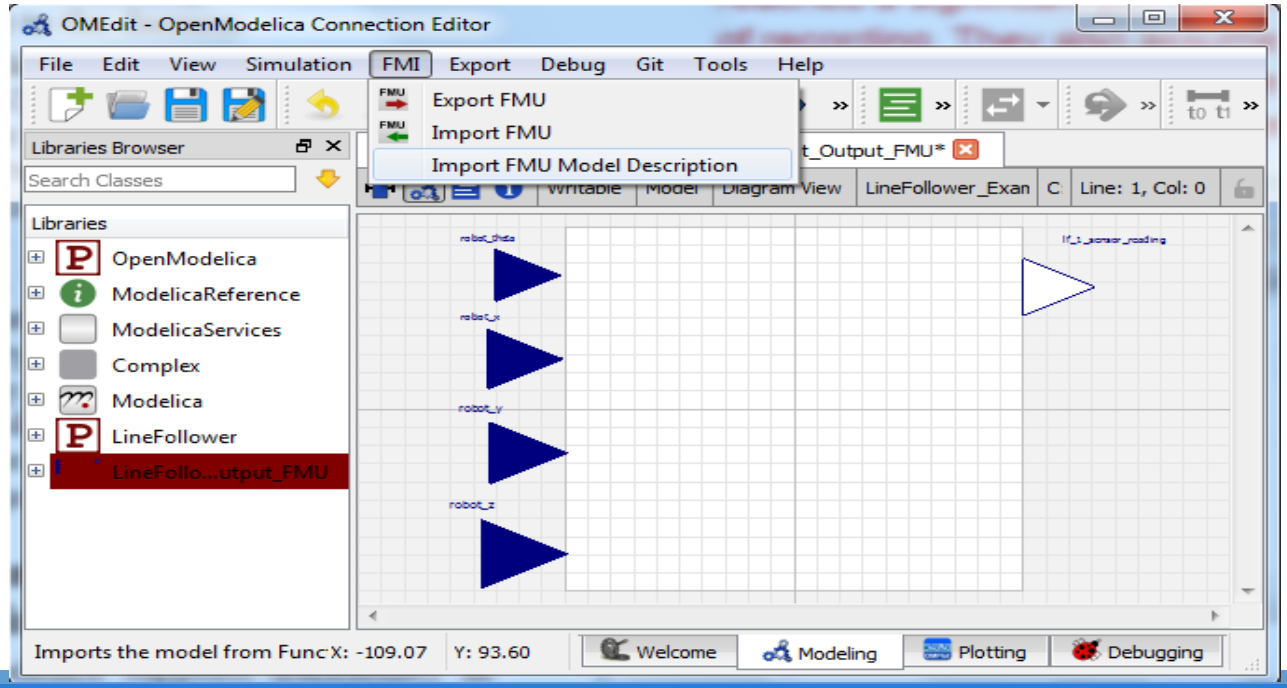


Prototype Implementation

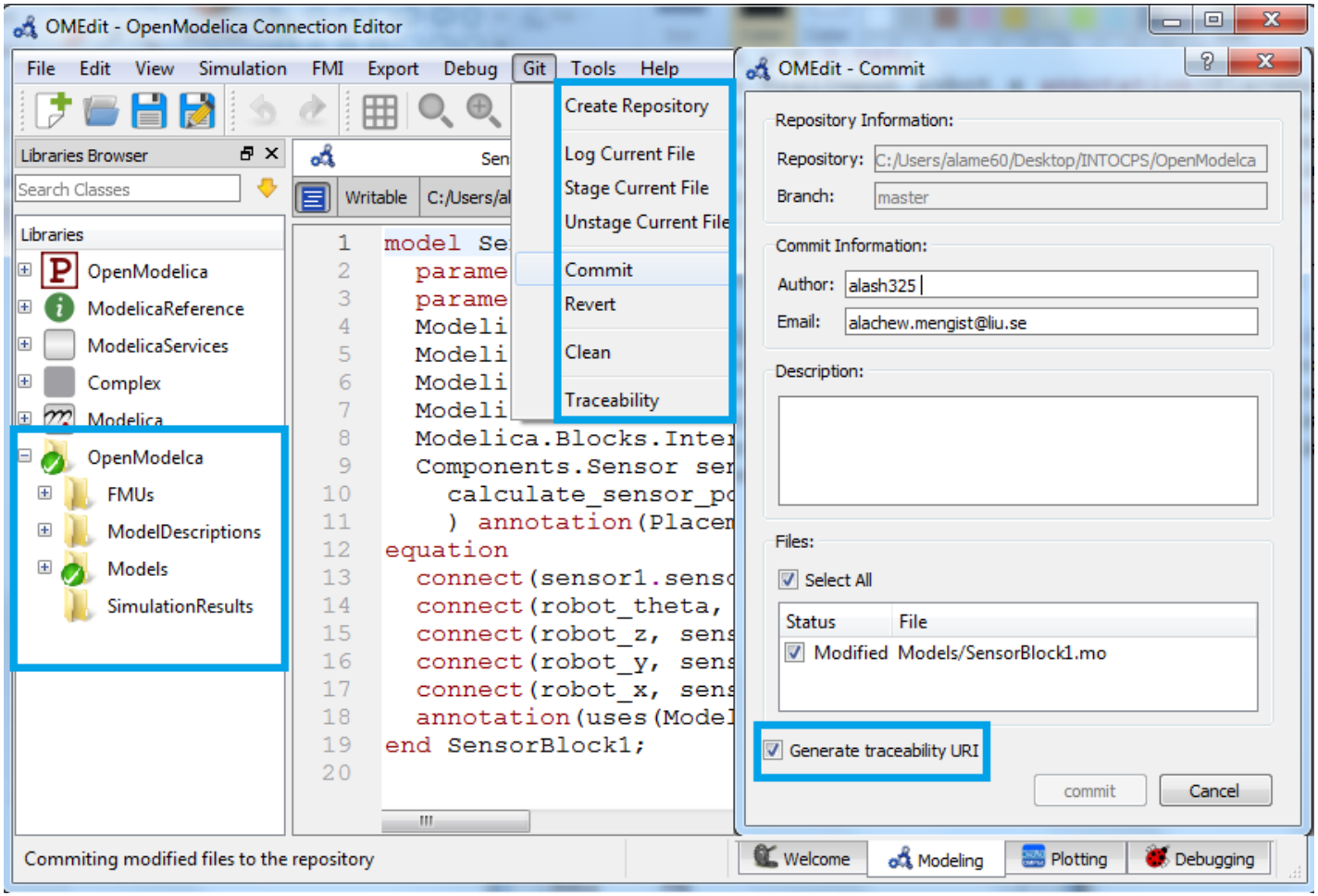
- An extension of the OpenModelica Connection Editor (OMEdit)
  - Implemented in C++ using the Qt graphical user interface library
- The prototype functionality into three categories:
  - Importing model description XML
  - Model management with Git integration
  - Traceability support using OSLC

Import Model Description XML File

- Import model description XML interface files (linked with requirements)
- Create Modelica model stub containing the inputs and outputs specified in *modelDescription.xml*

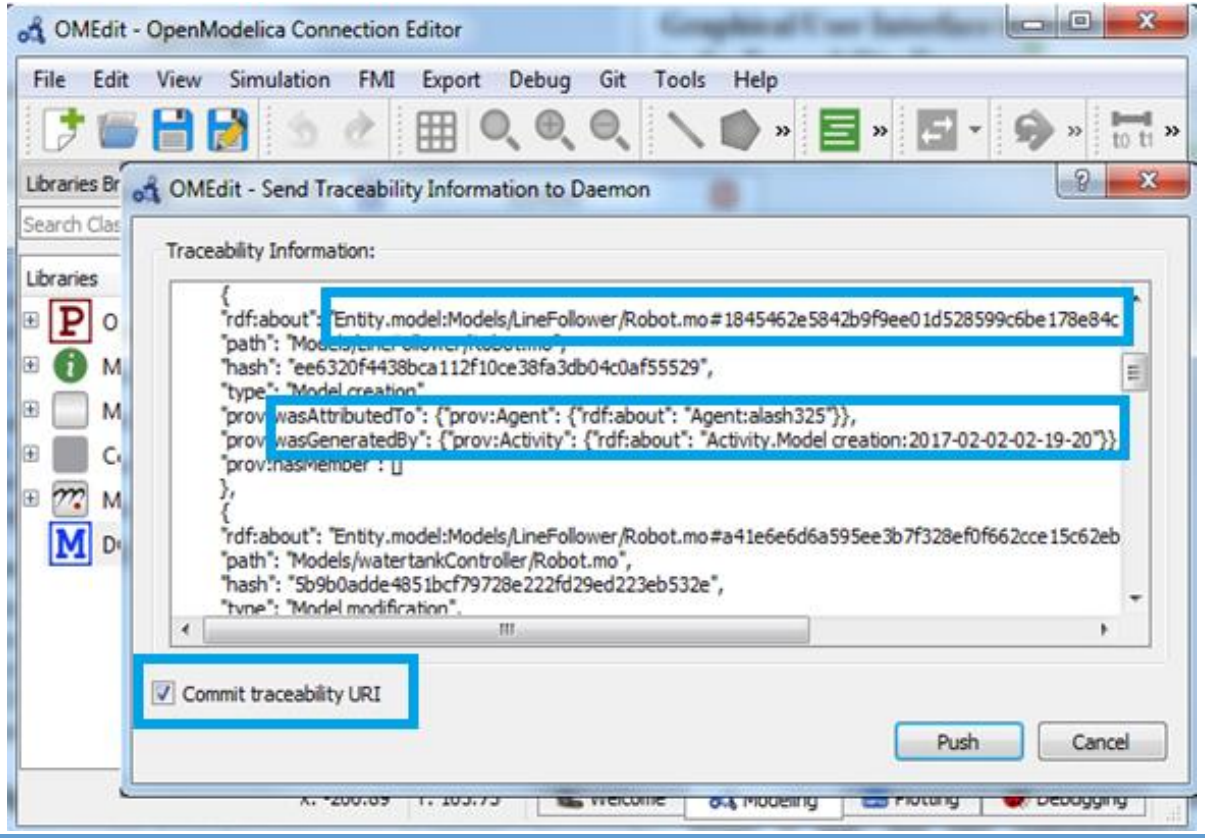


Model Management with Git Integration



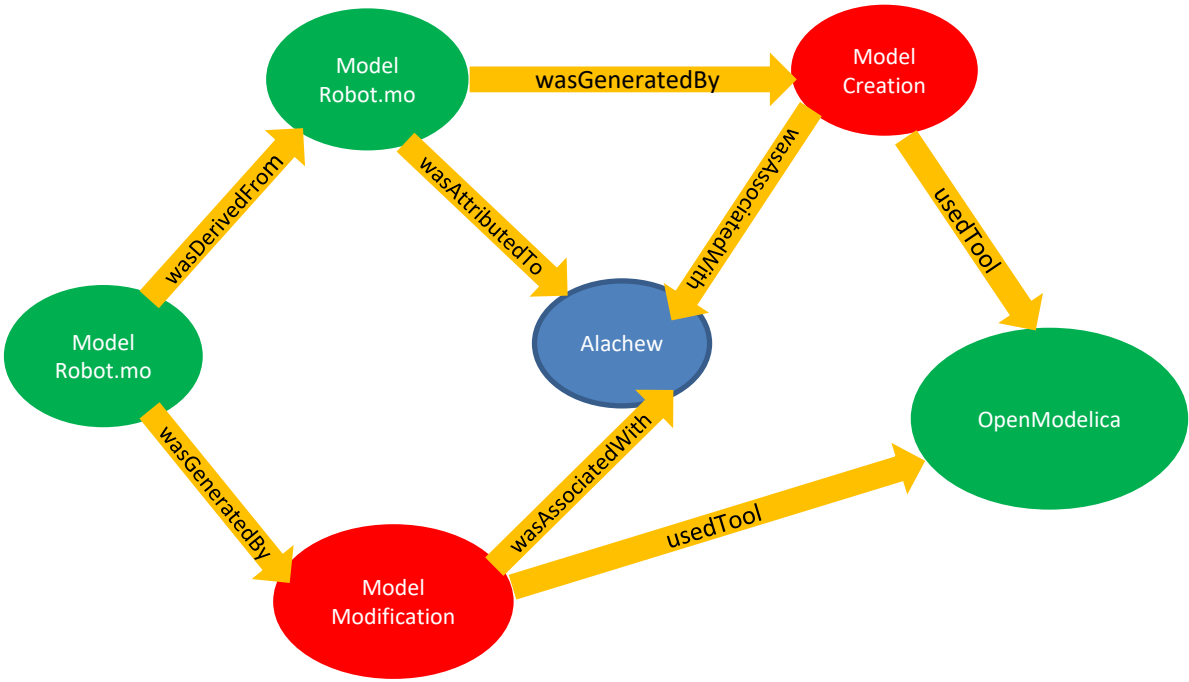
Traceability Support in OpenModelica

- Sent traceability information through the daemon to the database via HTTP POST
  - http://localhost:8080/traces/push/json



Traceability Support in OpenModelica

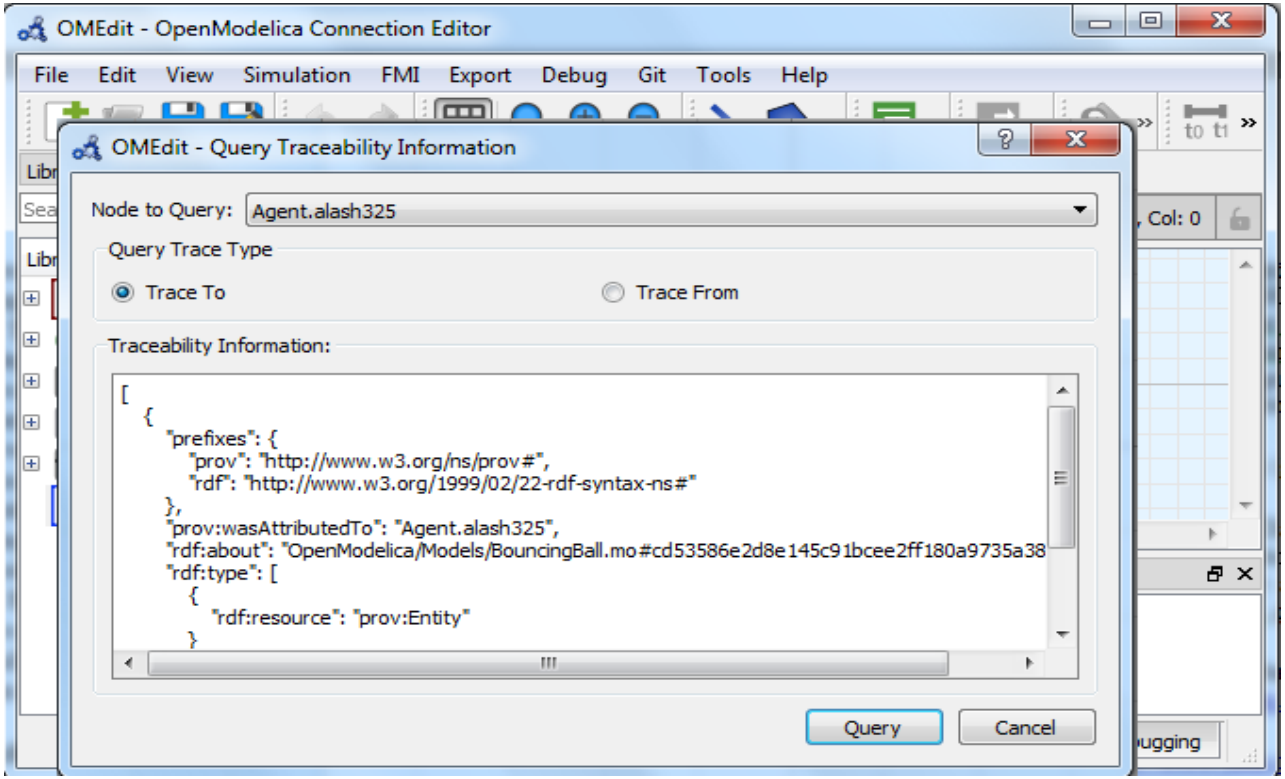
- Entities (e.g. `Modelica files, FMUs`) are shown in green
- Activities (e.g. `Model creation`, `Model modification`, `FMU export`) are shown in red
- Agents (e.g. a user with the name "`Alachew`") are shown in blue
- Their relationships (e.g. `wasGeneratedBy`, `wasDerivedFrom`, `usedTool`, …) are shown in orange.



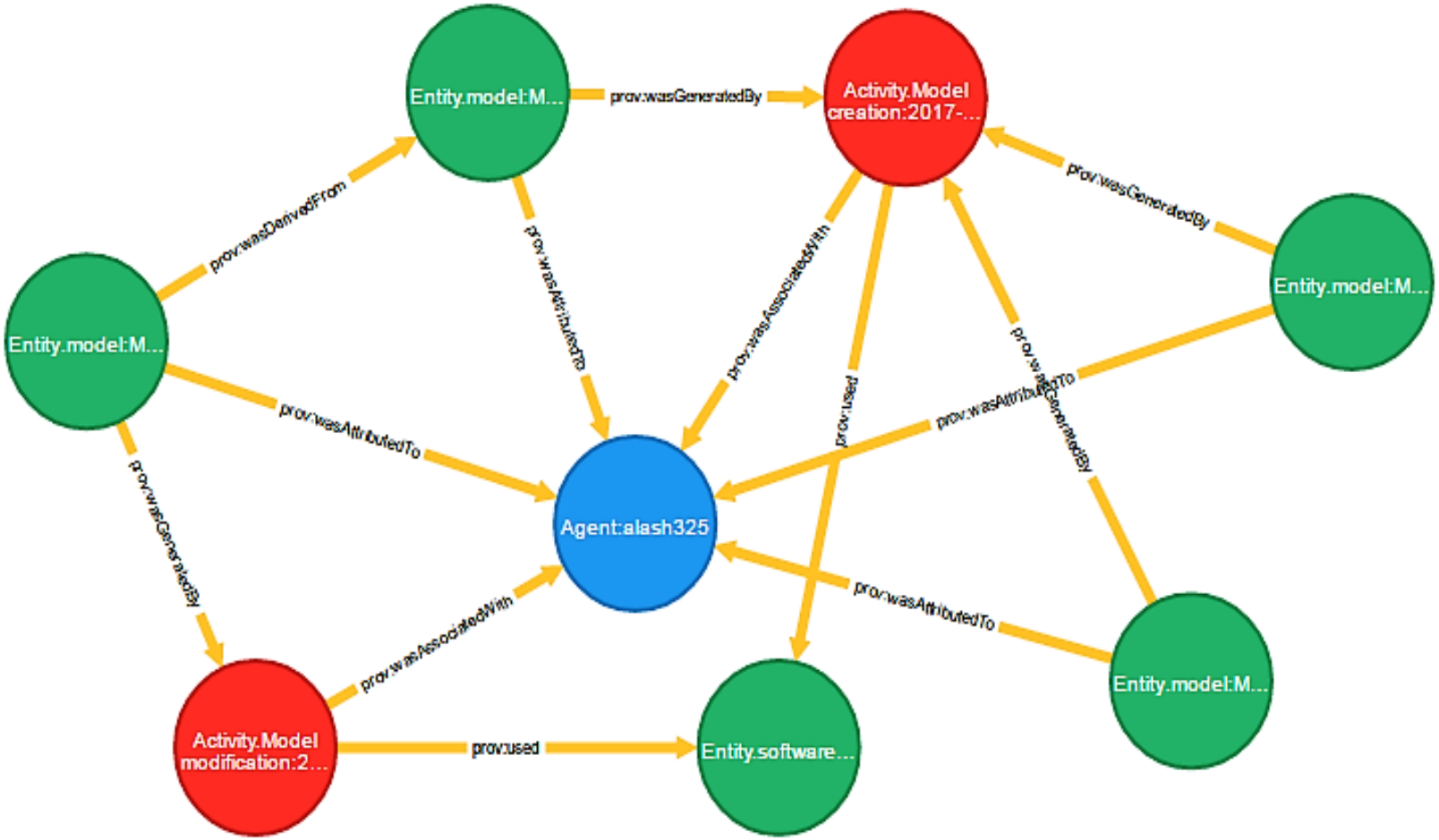
An example of traceability information sent from OpenModelica to the daemon and visualized in the Neo4j database

Traceability Support in OpenModelica

- Query traceability information (via HTTP GET)
  - Traces to (http://localhost:8080/traces/to/<URI>/json)
  - Traces from (http://localhost:8080/traces/from/<URI>/json)



The traceability database



Conclusions

- OpenModelica supports traceability Using OSLC
- The Modeling activities that can be recorded automatically within OpenModelica and traced are
  - Creation of models
  - Modification of models
  - Destruction of Models
  - Import of model descriptions in XML
  - Export of FMU's
  - Creation of simulation results
- A first prototype to query traceability information (traces to and traces from models or simulation results) from the database and display to end-users in JSON format is also complete.

Future Work

- Extend the OpenModelica tool to support visualization of the traceability data both in the form of graphs and trees.
- Fully functional Git integration
- Computing the impact of two different versions of the same model on simulation results and merging the models in way that the resulting model can be valid without modification.

Thank you for your attention!

Questions?