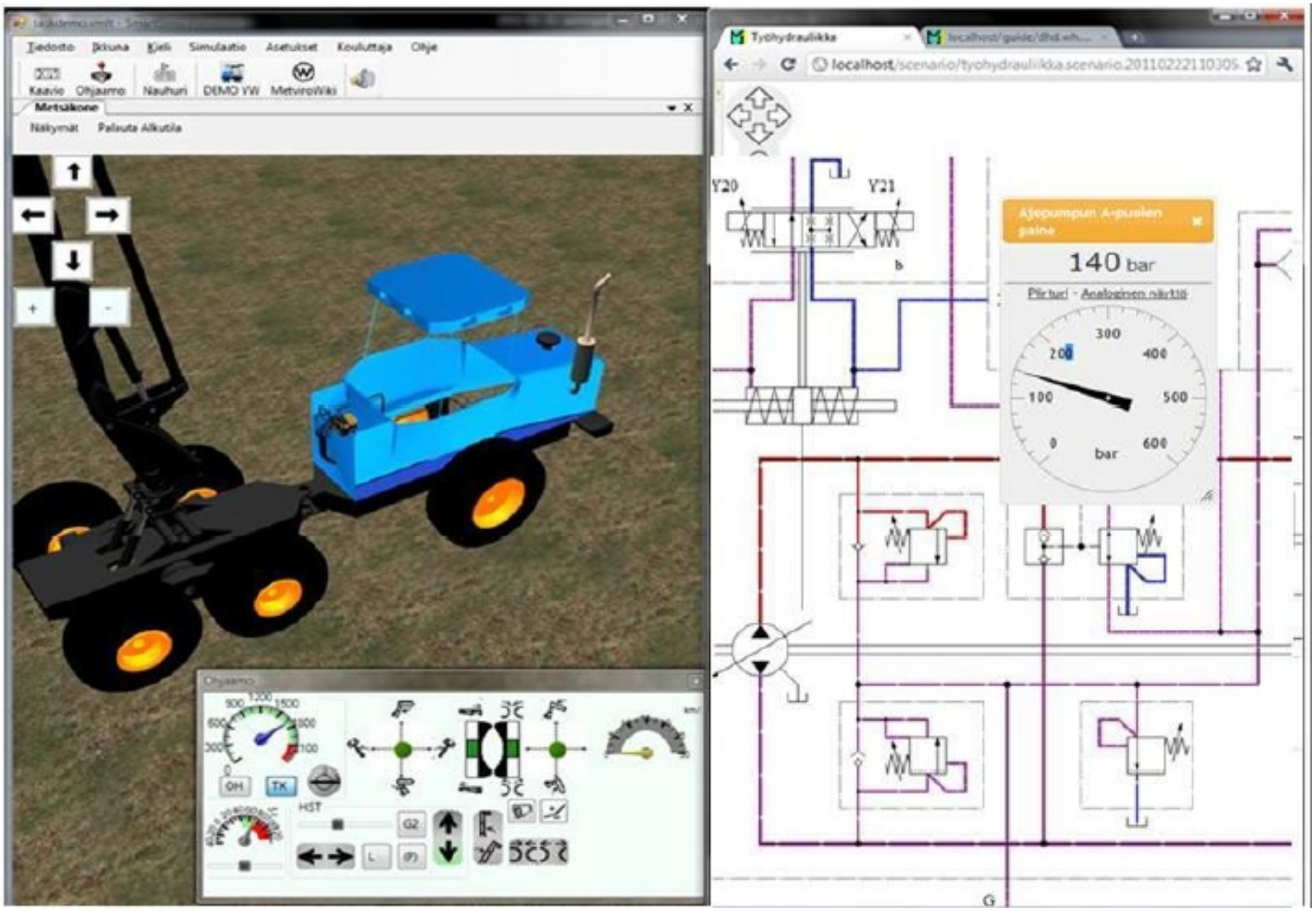
An Implementation of a Semantic, Web-Based Virtual Machine Laboratory Prototyping Environment

Jaakko Salonen
Researcher
Tampere University of Technology
Hypermedia Laboratory / Smart Simulators Research Group

In collaboration with: Ossi Nykänen, Pekka Ranta, Juha Nurmi, Matti Helminen, Markus Rokala, Tuija Palonen, Vänni Alarotu, Kari Koskinen, Seppo Pohjolainen

Virtual Machine Laboratories (VMLs)



Virtual Machine Laboratories

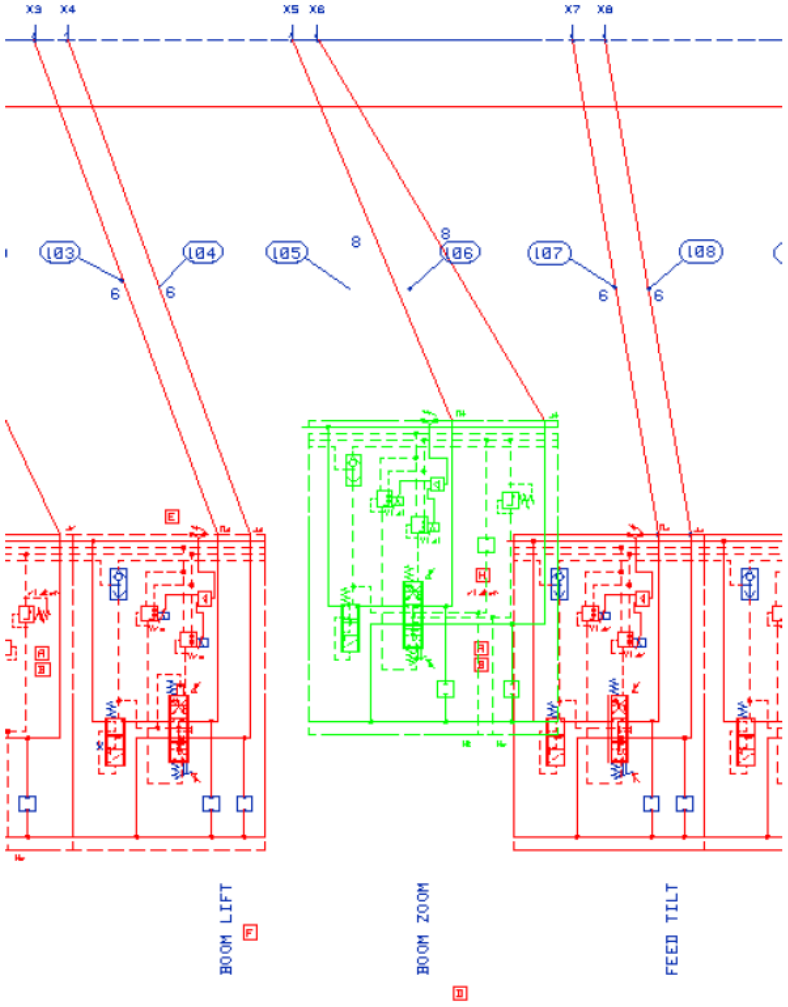
- **Virtual machine laboratories (VMLs)**
  - simulated planning and learning environments demonstrating function and structure of working machines
- Potential benefits for the industry
  - Design support (data integration, simulation)
  - Educational use (as a learning environment)
  - User guide (interactive maintenance and spare parts guide)
- Ideal case: complete virtual prototype
  - Allows iterating design without physically building prototypes

→ Cost-effectiveness, faster design evolution

Challenges

- VML development is non-trivial
  - Understand and codify structure and function [of a machine]
- Scattered, heterogenous design information
  - CAD drawings
  - 3D models
  - Documents, spreadsheets
- Document-based data
  - Informal
  - Silos of information (not interconnected)
- Undocumented, implicit design information
  - *”Yeah of course you can lift stuff with that we designed it for lifting”*
  - *”That's trivial, its the default value - 1.2561”*

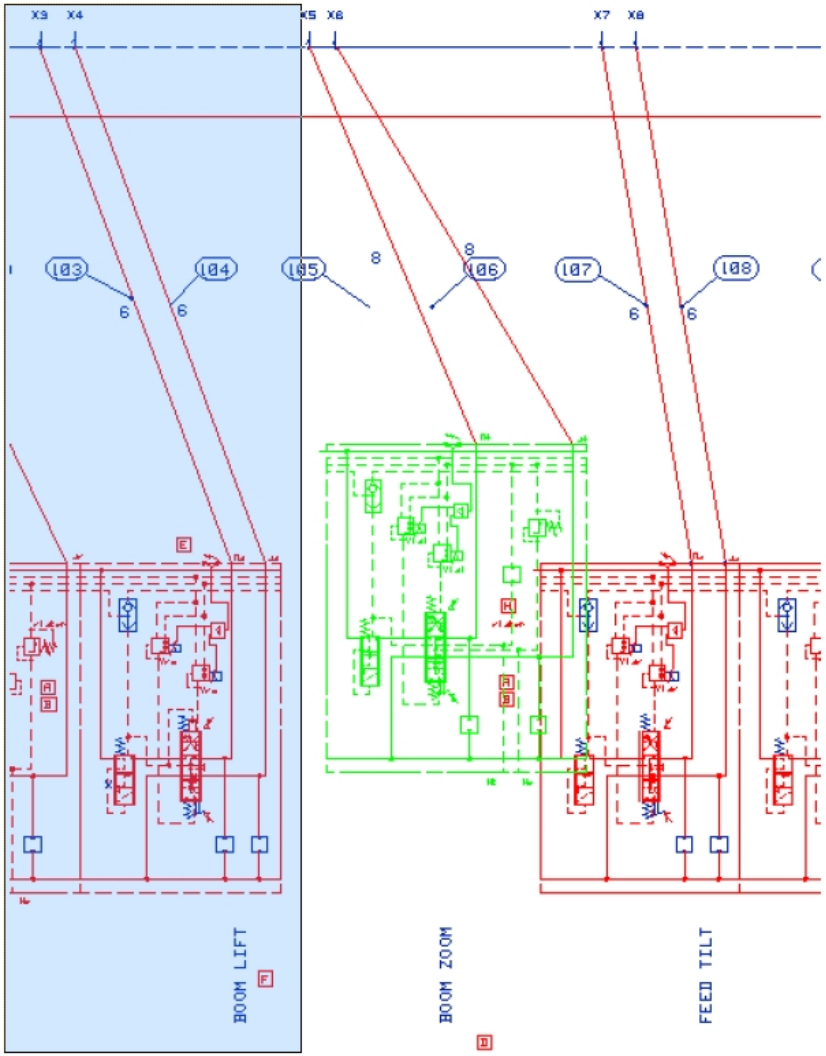
Designed for Visual Interpretation



”We all know that ” - Implicit Assumptions

Implicit assumption

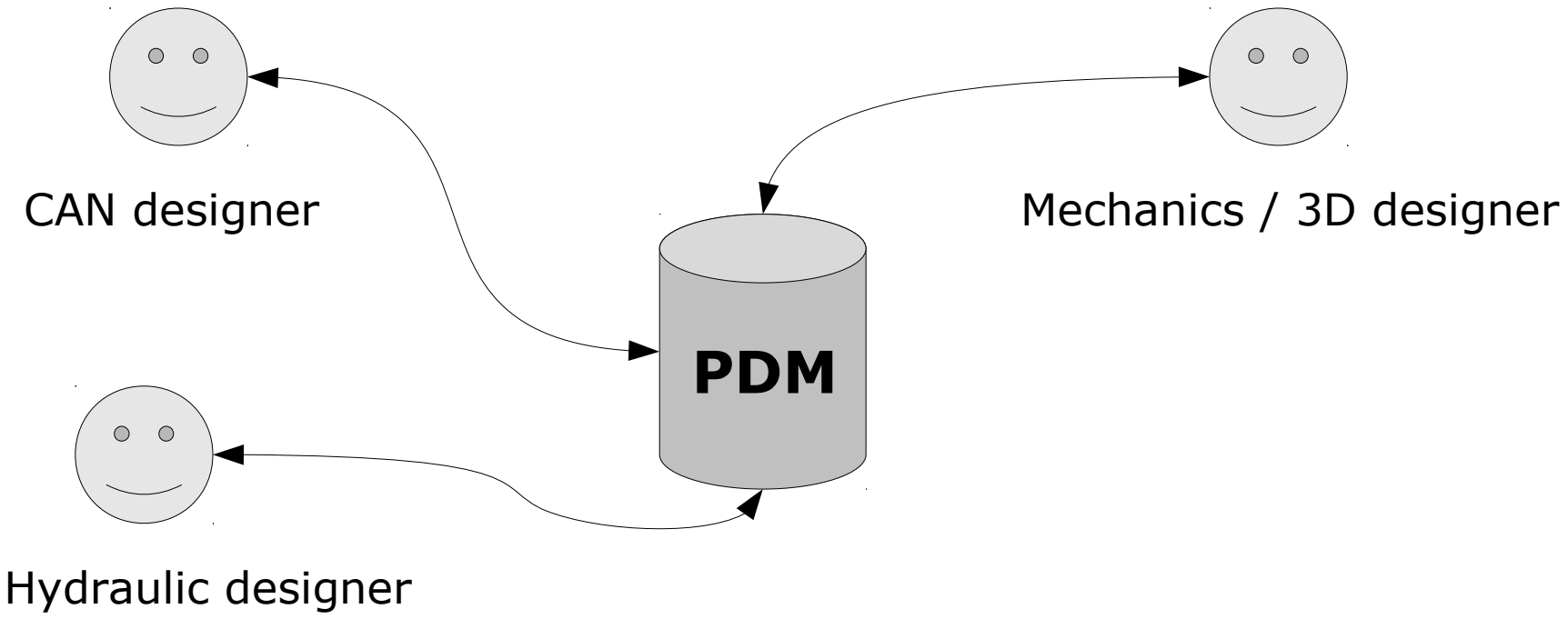
This part of the design implements the hydraulics for ”Boom Lift” functionality



Our Approach

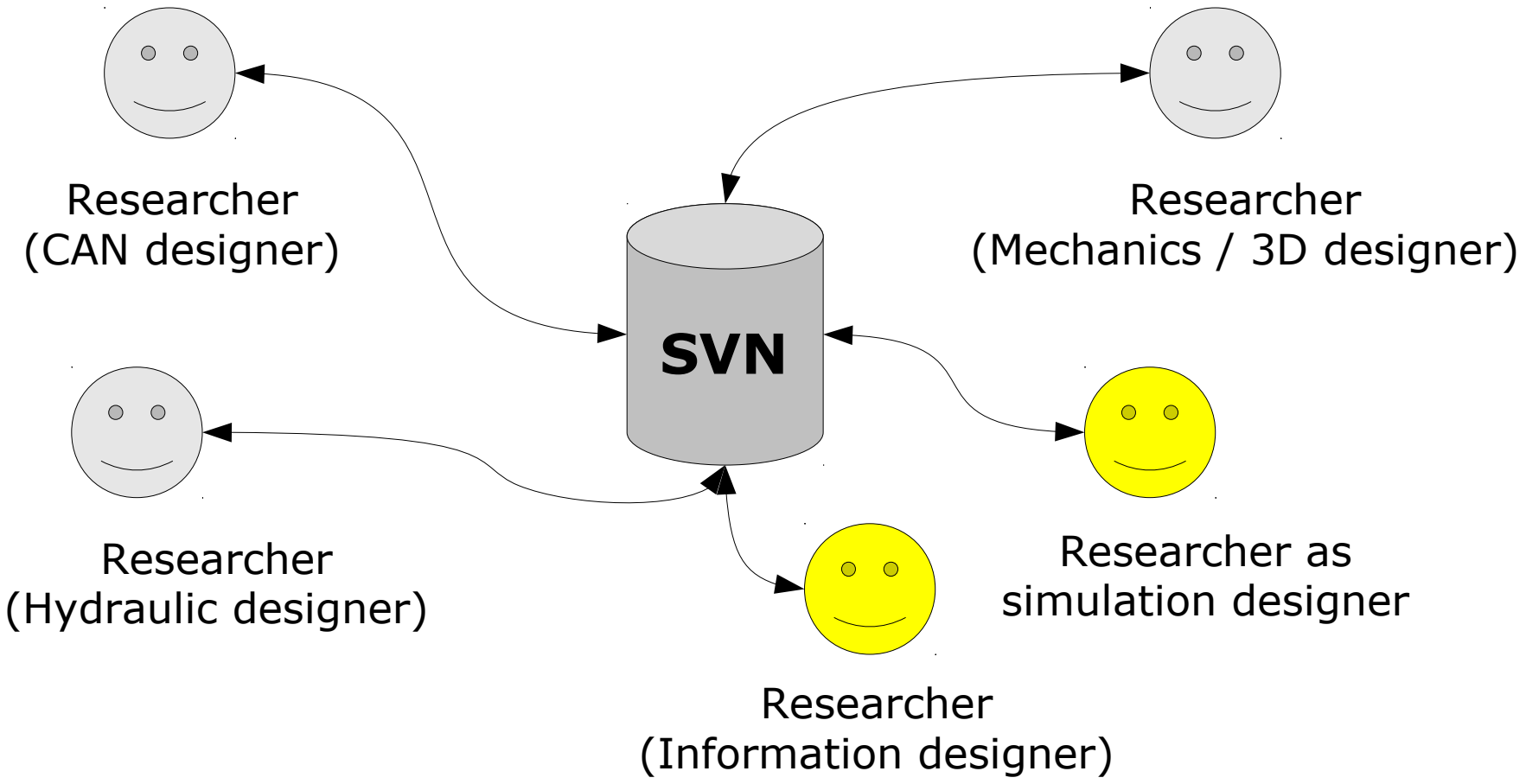
- Start with real design data, enrich and fix it
  - Structured, machine-readible design documents
  - Connect documents/designs with semantic links
- Minimally interfere with current workflows of machine design professionals
  - Attempt to simulate the way engineers collaborate on a machine design
  - Work in iterations
- Design the process and the environment in parallel
  - Roll-out new features as necessary

Assumed, ”real-world” Setup



→ Document-oriented

Our Simulated Setup

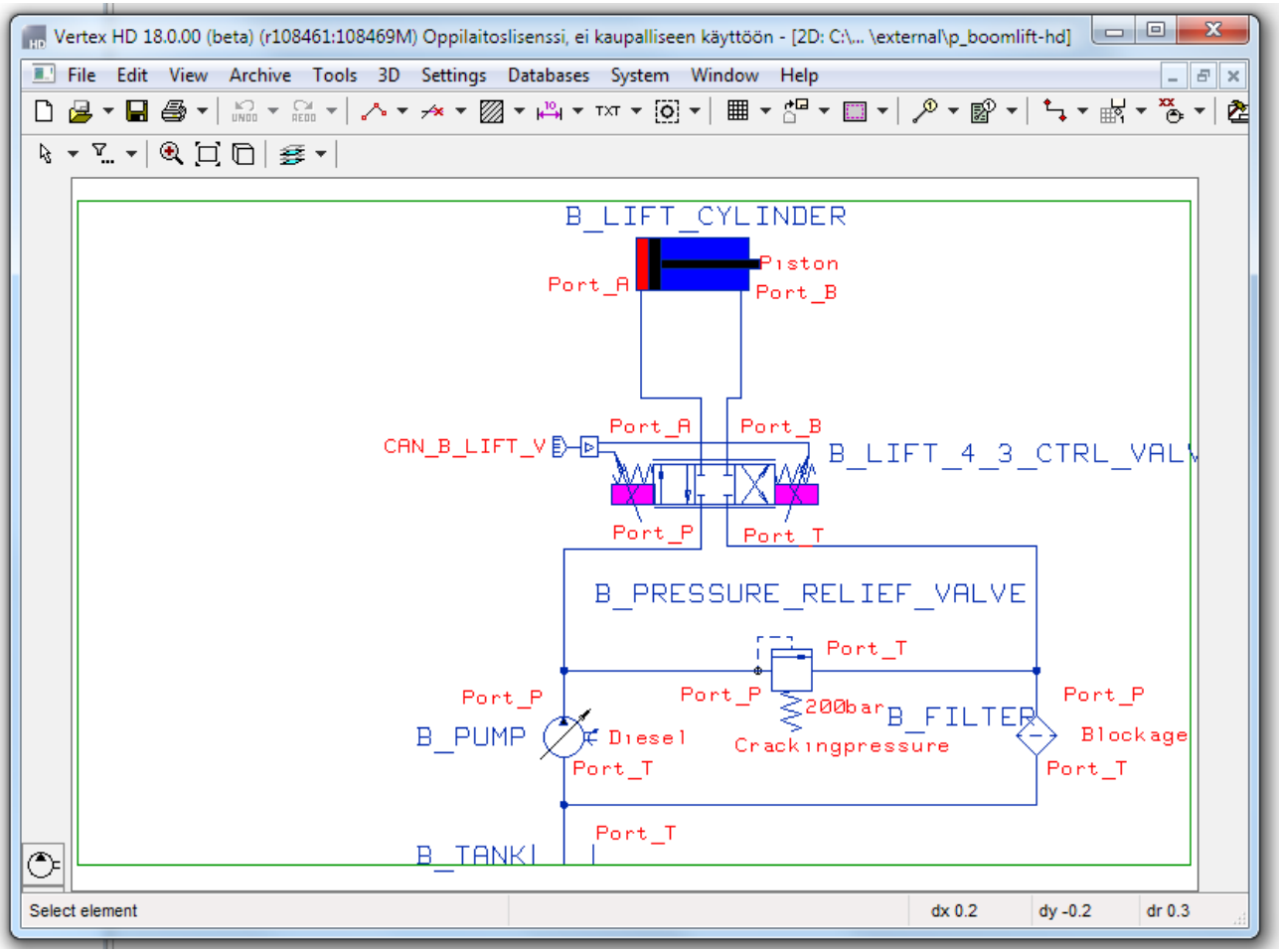


→ Structured Documents, Semantic models

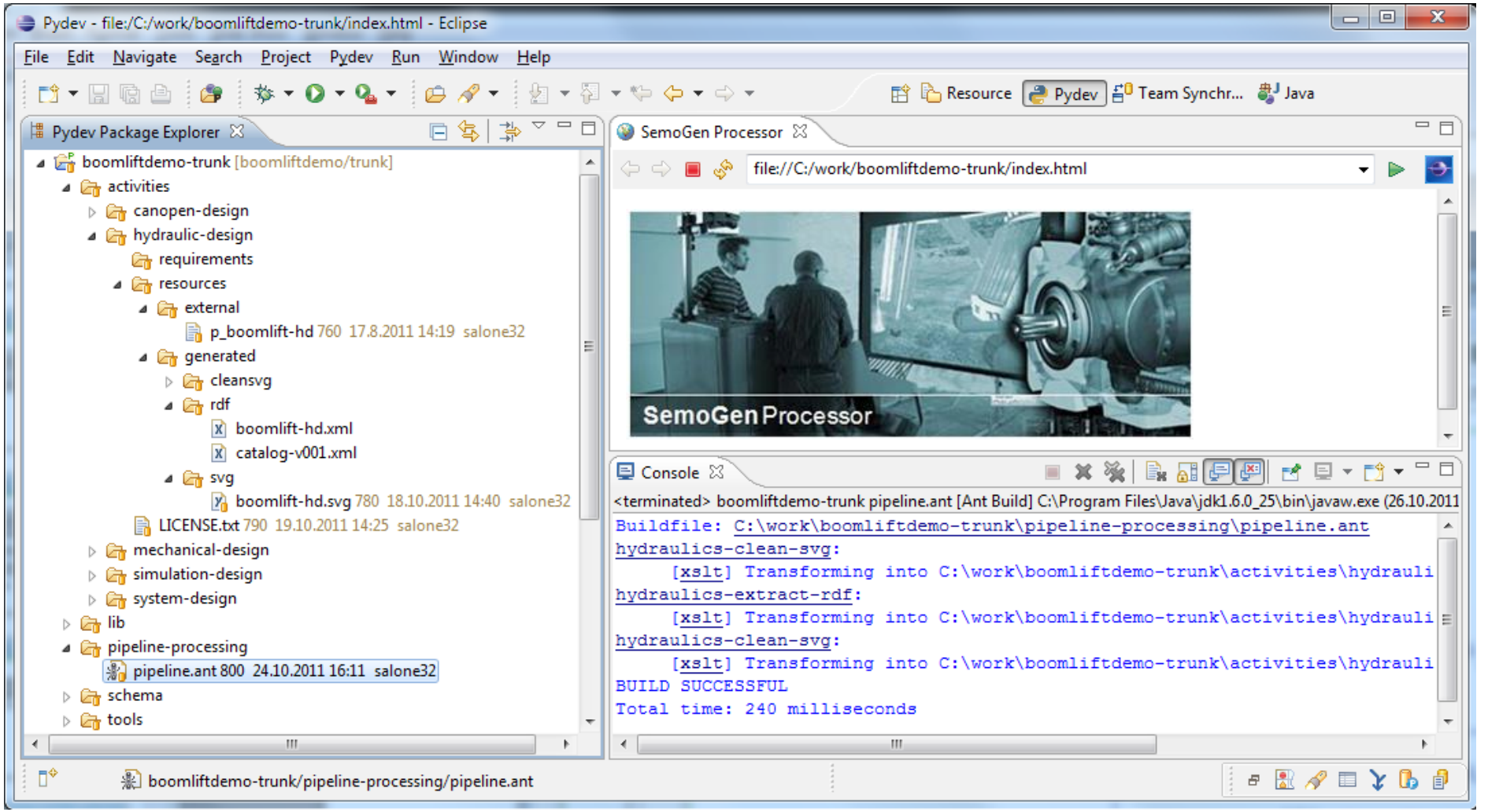
Environment

- Re-use original design applications as much as possible
  - Use CAD applications for data enrichments → hard-coded RDF only as a fallback
- Transform original designs to structured documents
  - Use export formats, adapters and APIs to create XML, SVG and RDF representations
- Create a toolchain from original designs into an integrated semantic model
  - Automated workflow and data aggregation
  - Help the designers with data validators
- Create a semantic viewer application that implements rudimentary virtual machine laboratory features
  - View 2D/3D designs, semantic search, real-time simulation support and measurements

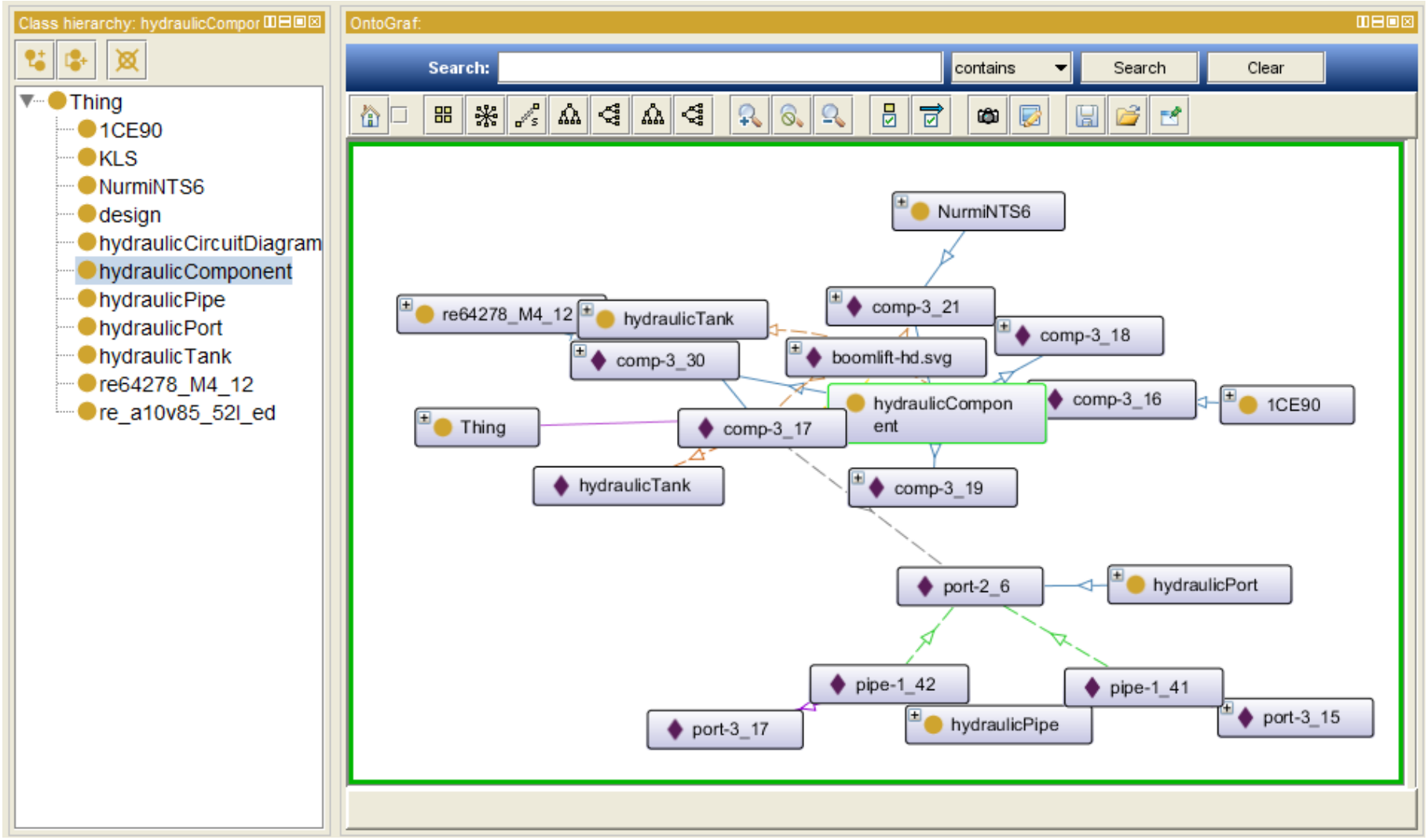
Walkthrough - Hydraulic design (1/4)



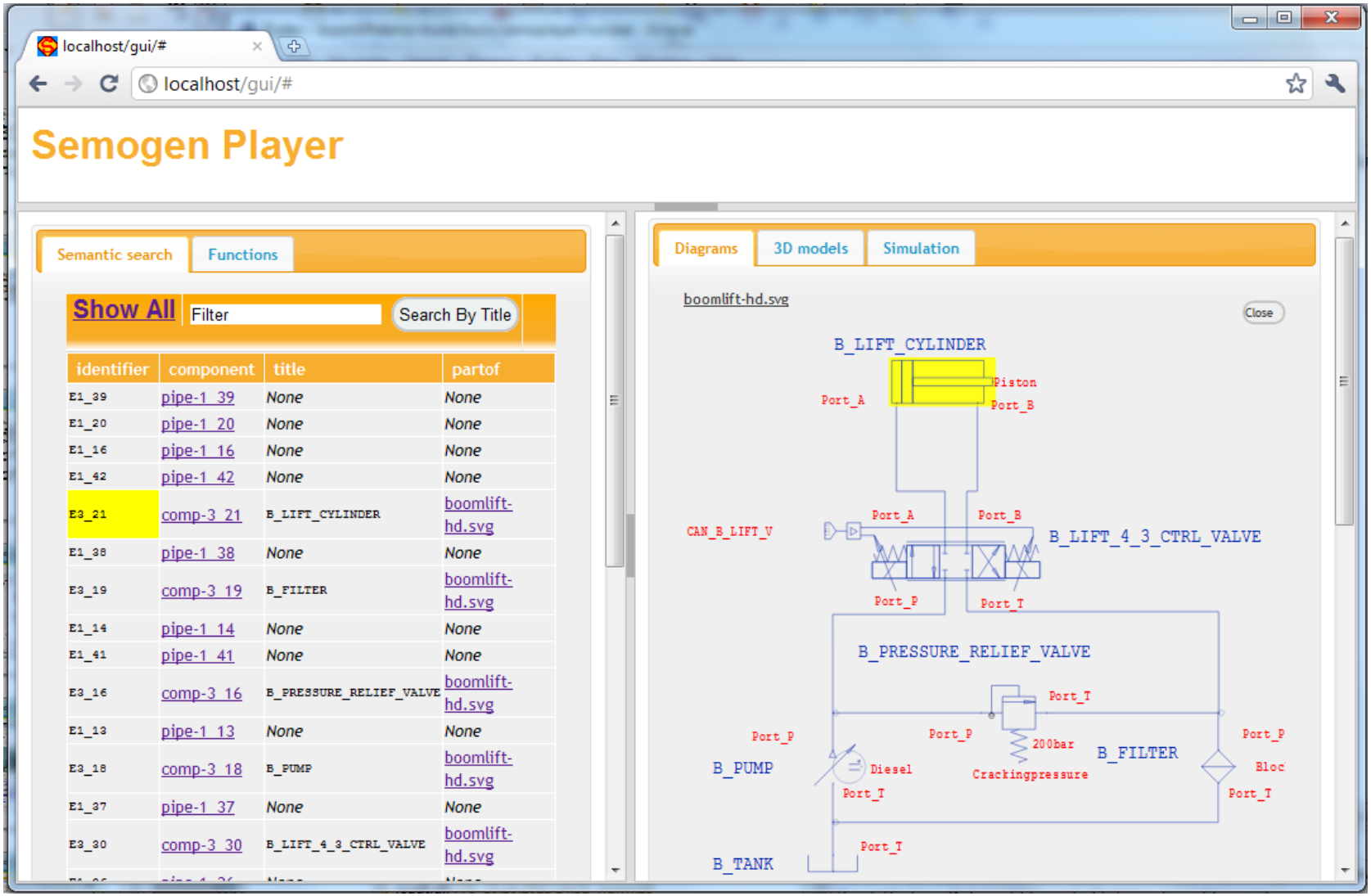
Walkthrough - Hydraulic design (2/4)



Walkthrough - Hydraulic design (3/4)



Walkthrough - Hydraulic design (4/4)



Results (1/3)

- **Enriched design documents**
  - CANopen design (3 networks)
  - One hydraulic design
  - One mechanical design
  - Cross-references between the given designs
  - An integrated simulation model
- **Process and Environment**
  - Implemented using Eclipse Platform along with some specific conventions and plug-ins
  - Workflow was automated with Apache Ant
  - Standard XML tools were used for RDF processing whenever possible by utilizing a canonical XML serialization

    → enabled to use Schematron validators for the RDF
  - Actual adapter components were implemented using XSL and custom Python scripts

Results (2/3)

- **Semantic Model and Ontology**
  - Started with ”only RDF”
  - Ended up using RDF Schema along with some OWL features (notably *owl:inverseOf*)
  - Created and mapped data against a newly created *semogen* ontology:
    - With models for *hydraulic*, *mechanical* and *CAN design* domains as well as *top level* models for integrating the domains
    - 52 classes, 58 objects
  - Data from design documents were mapped against the ontology
    - 618 instances

Results (3/3)

- **Semantic Viewer Application**
  - Implemented to create a quick (rough) VML prototype directly from semantic model (RDF)

    → no manual work required - only a model update is needed to refresh the viewer
  - Ended up using an open source stack (Python + JavaScript):
    - Tornado Web Server
    - RDFLib 3 + SuRF
    - jQuery, jQuery UI and jQuery UI.Layout for user inerface
    - X3DOM for 3D models

Discussion (1/2)

- Rather than creating the data model by firstly designing a normalized schema, we created the environment from features rising from actual design documents and processes.
- In terms of VML features, our environment is already able to provide the rudimentary aspects of the required features
  - Notably: rapid prototyping with evolving data, semantic search, simulation support
- Scaling to full design data is still a matter to be investigated
  - Technically doable, but design practices may need to be re-thinked – especially mapping components between designs is worksome → some automated tool may be required
- New design domains can be introduced as required
  - ...But requires developing new adapters

Discussion (2/2)

- In terms of technical problems, semantic web technologies weren't the bottleneck
- Most of the [technical] problems emerged due to usage of yet unstandardized browser features (WebSocket, WebGL)
- Notable problems arose from the application domain
  - Data encoded in a heterogenous set of documents
  - Roughly 100% ”Level 0” Linked Data
  - Largely lacks well-defined, open vocabularies, schemata and ontologies
  - Lack of explicit and open licensing of information our designs depends on (e.g. datasheets from component manufacturers)

Conclusions

- Semantic Web technologies were successfully applied to model virtual machine laboratory data
  - Enabled us to create a global, comprehensive representation of all the design data
- Complexity arising from heterogenous process and modeling environment is a notable challenge
  - In this flavor, adapters are a trade-off
- For wider deployment within the working machine industry, we would need to develop standard domain vocabularies and ontologies (as well as entity data!)
  - A more open, collaborative process for developing these artifacts

Thank you!

Jaakko Salonen (jaakko.salonen@tut.fi)

The prototype environment will be released under an open source license at **`http://wiki.tut.fi/SmartSimulators/`**

Description of the canonical XML/RDF serialization available at **`http://wiki.tut.fi/Wille/RDFcXML`**