

Visual and Secure Model Building in Risk Analytics



PillarOne Community Event, September 2nd, 2011

Martin Melchior, FHNW Windisch



- CTI Project: Usability of RiskAnalytics
- Models in RiskAnalytics:
What to specify and how was it done in the past
- Model Editor (Demo):
New Model Building Capabilities in RiskAnalytics
- Discussion and Next Steps

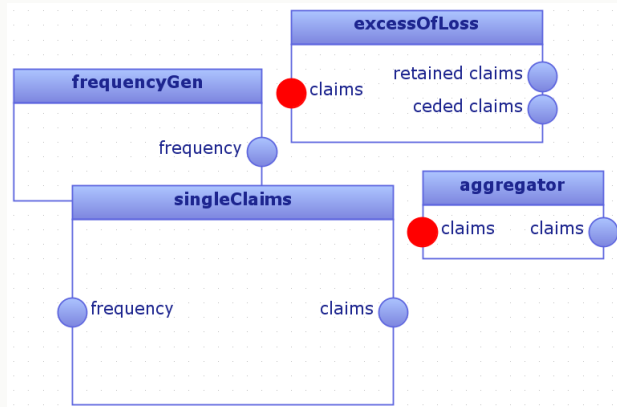
„Efficient building, computing and analyzing risk models for insurance companies with PillarOne“.

Joint project between Intuitive Collaboration AG and FHNW, funded by Swiss Government through CTI (Commission for Technology and Innovation), Intuitive Collaboration AG and Allianz Risk Transfer AG.

Three project phases:

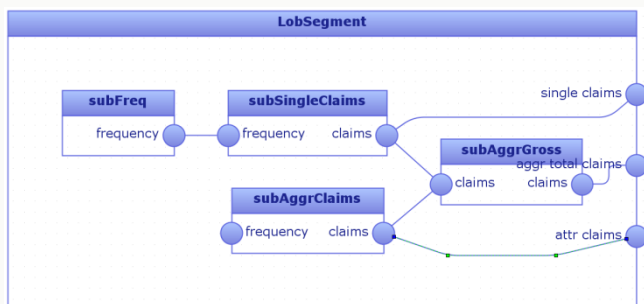
- ***Grid Computing Infrastructure:*** see presentation “Scalability of Computing Intensive Models” at the PillarOne Conference in 2010.
- ***Model Viewer & Editor:*** Work in progress, first release by end of September. Development supported by Canoo AG.
- ***Result Data Analyzer:*** Conceptual work started, implementation in Q4/11.

- **Component-based** to manage the complexity



- Low-level components: Primary calculation logic (e.g. random number generators, mechanics of how re-insurance contracts work, etc.)
- Composed components: Combine lower level components to higher level functional components.

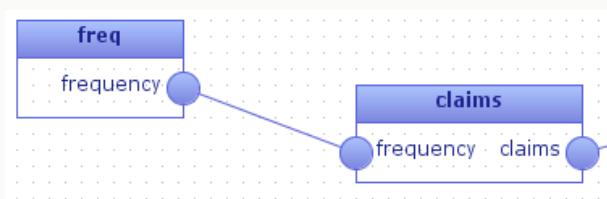
- **Nested model structures**



- Models are composed of components that may again be composed of components, etc.
- Can be represented by a model tree.

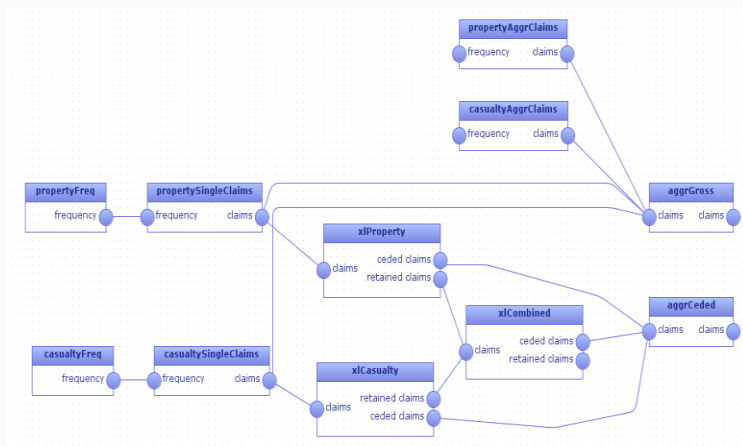
Connectivity of Models

- Data transported between components through connections.
- Connections plugged to components at suitable attachment points ('ports') → Define In-/Output from/to components.
- Exchange format are suitably typed 'Packets' (e.g. Claim, Frequency) → Type safety for connections.

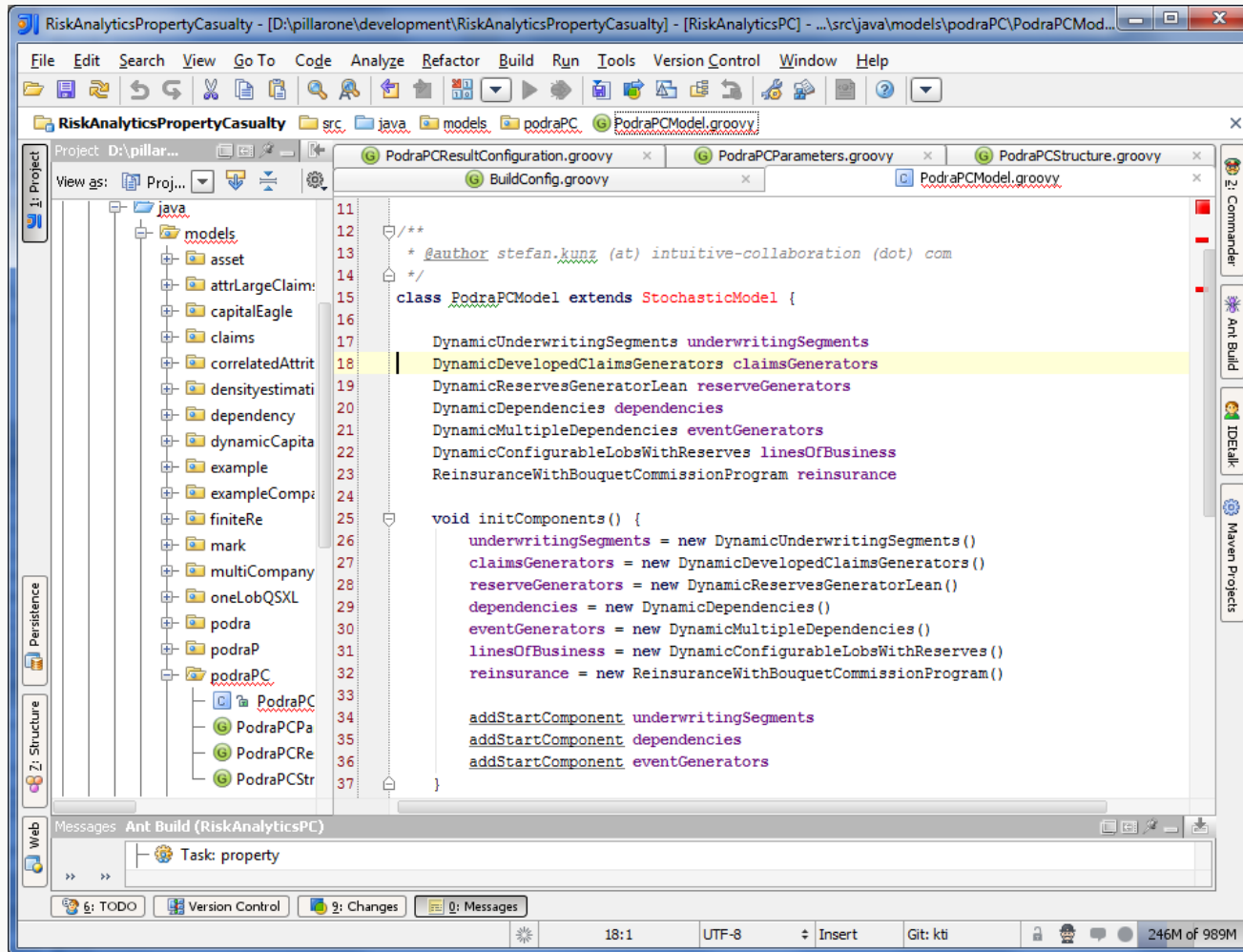


Data Flow Models

- Clearly identifiable starting points where the data flow needs to be initiated.
- Each component processes the data once all input data is available and sends it forward once done.



Models in RiskAnalytics (3)



Development of Models (past):

- Writing **groovy** code
- Using a suitable **integrated developer environment** that supports you in doing so.

High entry barrier for modelers:
Typically, not in the core competence of a risk modeler / actuary.

Modell Editor in RiskAnalytics: Transparency (1)

The image shows two overlapping screenshots of the 'Model Editor' application. The top screenshot displays a 'Tree' view of a model named 'ModelA'. The left pane shows a 'Components' list with items like 'aggr gross', 'aggr ceded', 'xl property', 'property single claims', 'frequency', 'claims', 'property freq', 'casualty aggr claims', and 'casualty single claims'. The right pane shows a 'Connections' table with columns 'From' and 'To', listing relationships between various claim types and aggregators. A large, semi-transparent text overlay reads 'Model Tree Connections'. The bottom screenshot shows a 'Visual' view of the same model, where components are represented as nodes in a network graph. Nodes include 'propertyAggrClaims', 'casualtyAggrClaims', 'aggrGross', 'propertyFreq', 'propertySingleClaims', 'aggrCeded', 'casualtyFreq', 'casualtySingleClaims', 'xlProperty', 'xlCombined', and 'xlCasualty'. Lines connect these nodes, illustrating the flow and dependencies between different parts of the model.

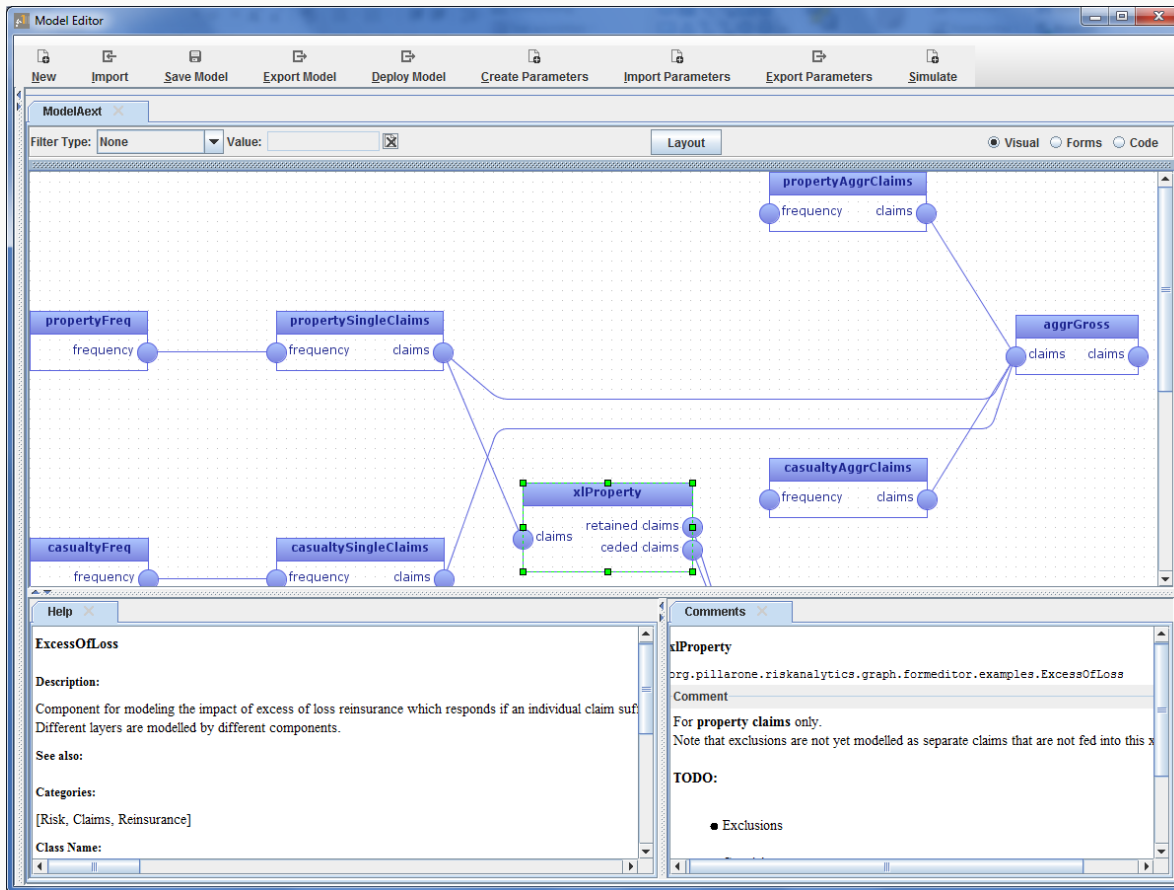
Different views allow to inspect given models or edit new models:

- **Visual:** Reveal connectivity of models. Automatic or manual (and saved) layouts.
- **Tree:** Reveal hierarchical structure of models.
- **Code:** Textual representation (groovy code)

Search/filter functionality:

- Different search filters: Categories, Packet types, Names, etc.
- Consistent across different views

Modell Editor in RiskAnalytics: Transparency (2)



Context-sensitive help

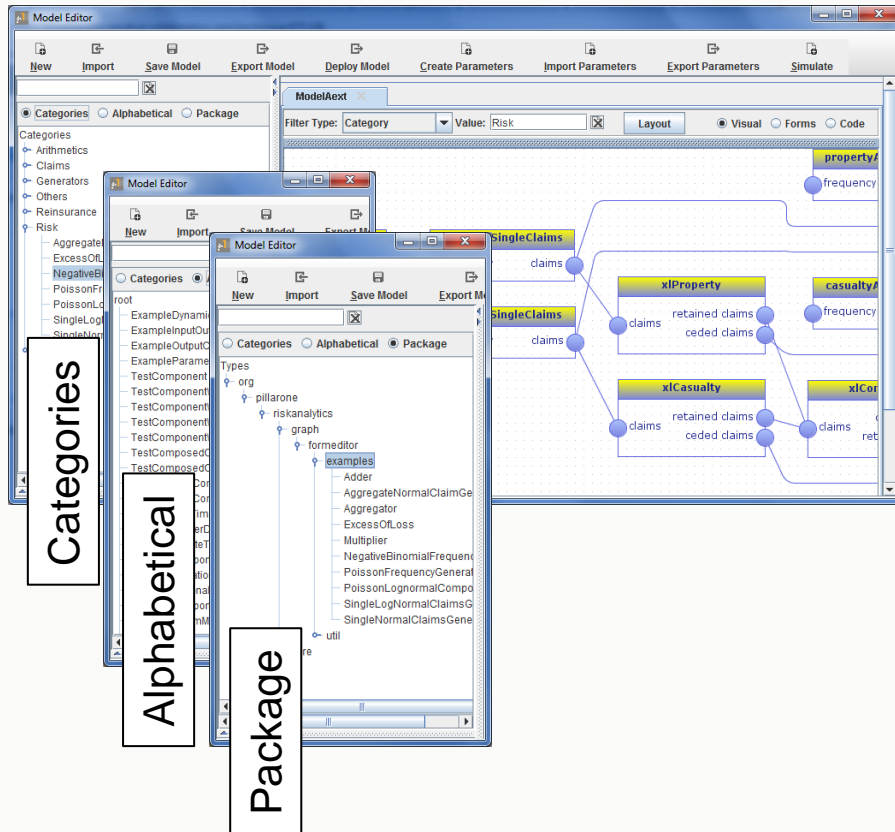
- Some parts automatically generated: always up-to-date!
- Context help as comprehensive as provided by the business logic libraries.

Comment editor

For specifying context help on models built by the user.

- html formatting
- persisted with the model

Modell Editor in RiskAnalytics: User Guidance

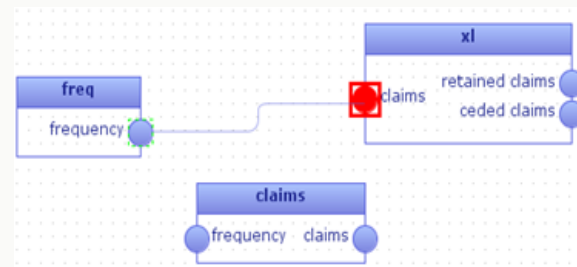
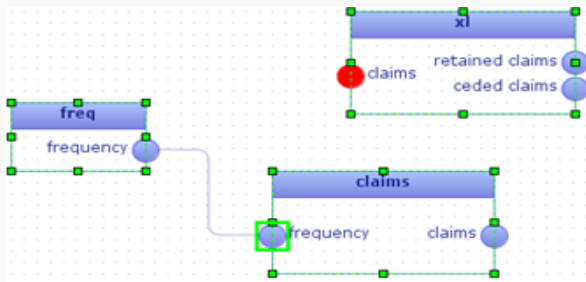


- Well organized component **palettes** to easily find the components.
 - Different views (Category, Alphabetical, Package)
 - Search/Filter functionality
 - Context help on selected items
- Model **editing area**
 - Different views
 - Consistent drag & drop from palette
 - Consistent and unique naming for all model items (components, ports).
 - Assistance for defining composed components.
- **Start-up screen** with hints on how you can proceed

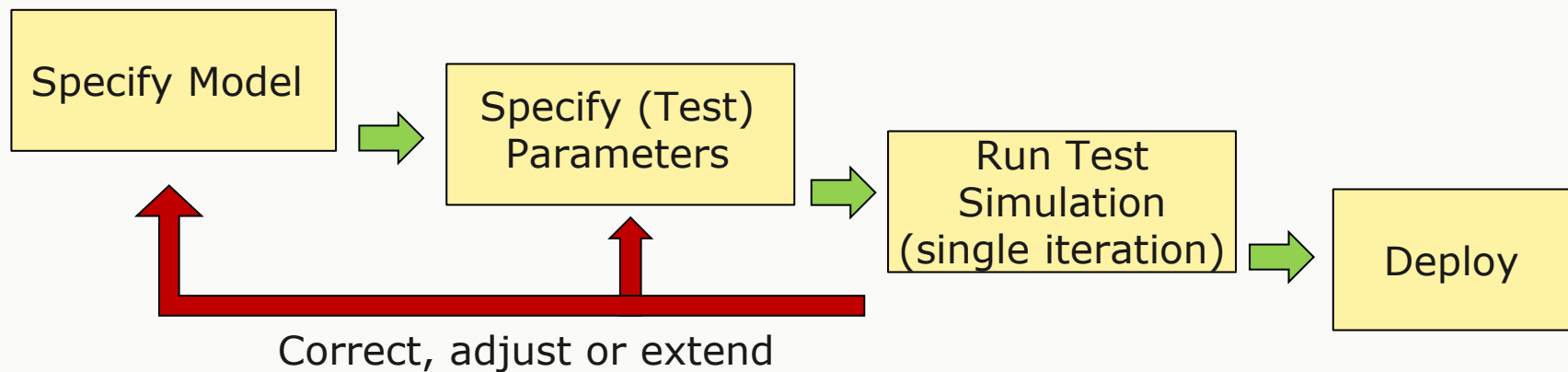
Modell Editor in RiskAnalytics: Model Validation & Testing

Data flow consistency by validation rules:

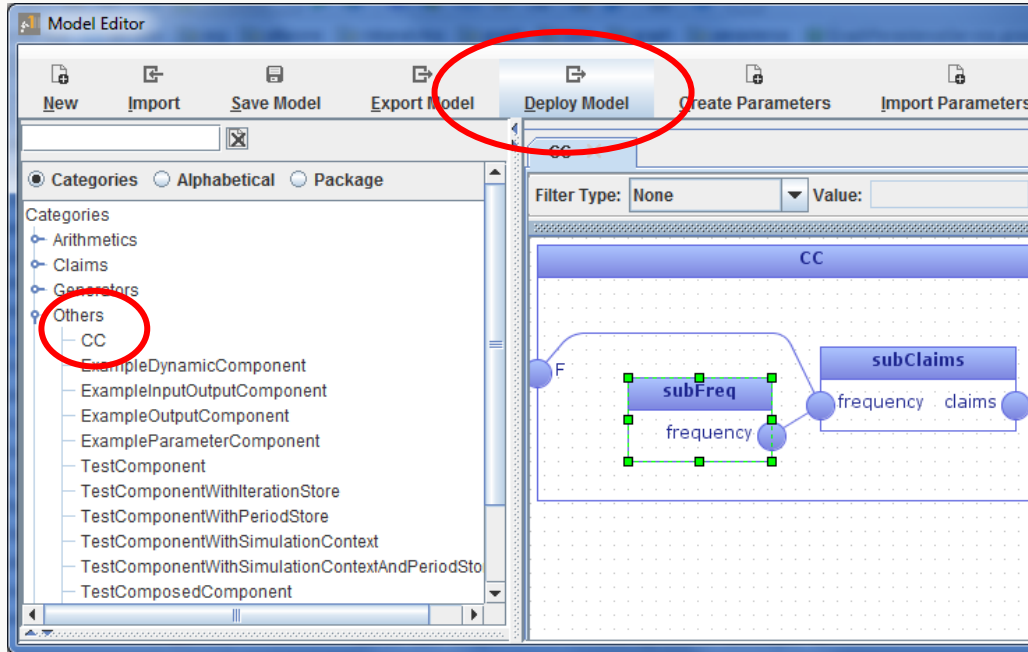
- Checking of packet type and number of connections attached to ports
- Dynamic information on permissible connections



Support for iterative model development:



Modell Editor for RiskAnalytics: Deployment



Dedicated deployment step

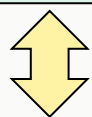
- Deploy tested models for simulations in RiskAnalytics.
- Deploy composed components for re-use in model building (added to palette) and in RiskAnalytics.

Full integration with RiskAnalytics (with all its benefits):

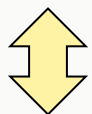
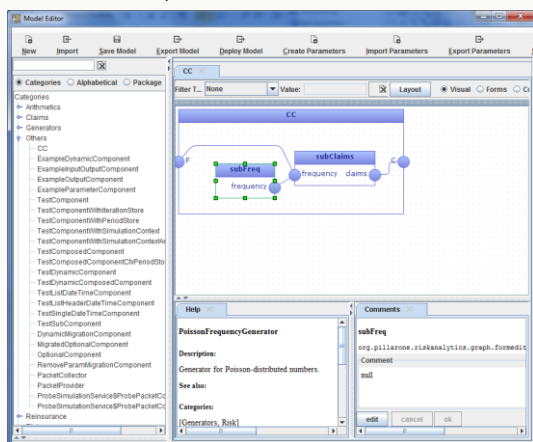
- Client/server architecture
- Full integration with DB system
- All data input / output functionality
- etc.

Model Editor for RiskAnalytics: Model/Data Exchange

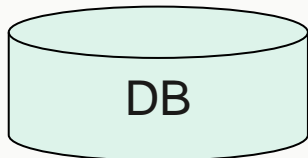
Groovy
Files



**Parameterizations
Models**



**Models
Layout Information
Parameterizations**



- Import / Export models: groovy code
 - Useful for exchanging models without using the DB on the server.
 - Inspect existing models.
- Save / Load models under development:
 - Before deployment to save work in progress.
- Import / Export parameterizations prepared in the Model Editor.
 - Develop test data in parallel to the model. Use the test data for operations in RiskAnalytics.
- Save / Load layout information:
 - For high quality pictures of the models for presentations.

- First Release: End of September 2011
 - We are almost there!
- Community support: Feedback / Work / Funding?
- Possible future extensions:
 - Scriptable component template ('white boxing'), possibly a GUI to assist the user in creating the scripting part.
 - Code editor
 - Show implementation code for simple components.
 - Model refactoring support
 - Versioning concept: Model versions
 - etc.

- A model editor for viewing and editing RiskAnalytics models and building blocks has been implemented and incorporated into RiskAnalytics.
- A visual, a tree and a code view are provided that are kept consistent at any time. The user can arbitrarily switch between the visual and the tree view to edit the model.
- Advanced model validation functionality is integral part of the editor: Validation rules govern how the user can connect components. At a higher level, test simulations can be run at any time during the model development.
- Model and sample dataset can be specified and tested in parallel, iteratively.
- Context-specific help allows the user to explore the models and components and assists him/her in what components to use.
- Filter and search functionality allow the user to drill the model.

Martin Melchior
University of Applied Sciences Northwestern Switzerland
Steinackerstrasse 5
5210 Windisch

+41 56 462 40 31

martin.melchior@fhnw.ch