Tool Demonstration

Henceforth, we describe and illustrate the installation and usage of FRaMED.

Installation

To install FRaMED, a user has several alternative options.

First and foremost, we have prepared a **virtual machine** for the virtualization environment VirtualBox that can be installed, as follows:

1. Download and install VirtualBox for your platform from https://www.virtualbox.org/
2. Download the FRaMED virtual machine from http://st.inf.tu-dresden.de/intern/framed/framed-ubuntu.ova *(1.8 GB)*
3. Open your VirtualBox and *Import* the downloaded file
4. Startup the added framed-ubuntu_1 virtual machine

Alternatively, we packed FRaMED into a **compressed** Eclipse **product** that can be installed, as follows:

1. Make sure to have Java Development Kit 7 and corresponding Java Runtime Environment 7 installed
2. Download the packed FRaMED product for your platform from https://github.com/leondart/FRaMED/releases/tag/v2.0.3
3. Unpack the archive and start Eclipse by clicking on the eclipse executable

Those, who want to inspect the plugins implementation, can find a step-by-step installation in the FRaMED wiki.[8]

First Steps

After, FRaMED is launched you can either import the example Project via GitHub or create a new Project.

For the former, select *File* > *Import* ..., select *Projects from GitHub* and clone the following repository at: https://github.com/Eden-06/FRaMED-Example. Within the example project, you will find two *.crom_dia files, that can be opened by double-clicking them in the package explorer.

For the latter, select *File* > *New* > *Java Project* and create a new project. Afterwards, you can add a new CROM model by clicking *File* > *New* > *Other* ... and selecting *CROM Role Model* within the *Role Modeling* category, as depicted in Fig. 4. As a result, the Editor UI opens the newly created empty file.

Modeling Role-based Applications

Once you have opened the graphical editor, you start in the top-level view of your application. Fig. 3 shows an example of the top-level view. Here, you can perform the following actions:

- Create natural, data, and compartment types,
- Specify their inheritance relation,
- Create and refine the fills-relation (q.v. Fig. 5), and
- Step into individual compartment types (q.v. Fig. 6).

Within a compartment type, the palette changes to reflect the different perspective, as depicted in Fig. 7. In detail, the following actions are supported:

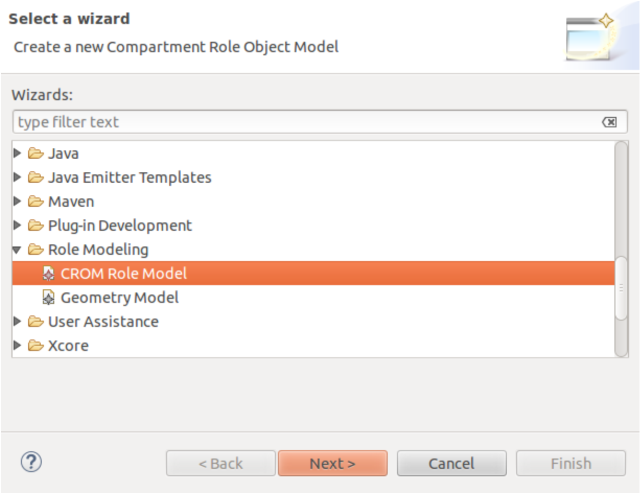

Figure 4: Creation wizard for CROM

- Create role types and role groups,
- Specify various role constraints,
- Create relationship types between two role types,
- Specify inter-relationship constraints,
- Add intra-relationship constraints (q.v. Fig. 8), and
- Step out of the compartment type (q.v. Fig. 9)

Last but not least, the code generation can be triggered by right-clicking on a *.crom file, unfolding the *Generate* menu item, and selecting one of the following options:

- Generate the *formal CROM* (*.py file), or
- Generate the *SCROLL source code* (*.scala file).

The former can by executed with the following shell command:

```
python YourExample.py.
```

This will validate the given model assuming that at least Python version 2.7.3 is installed and the crom.py library[9] exists in the same folder. For the latter, we refer the reader to the SCROLL documentation available on GitHub.[10]

[8] https://github.com/leondart/FRaMED/wiki/Step-by-Step-Installation

[9] https://github.com/Eden-06/formalCROM

[10] https://github.com/max-leuthaeuser/SCROLL

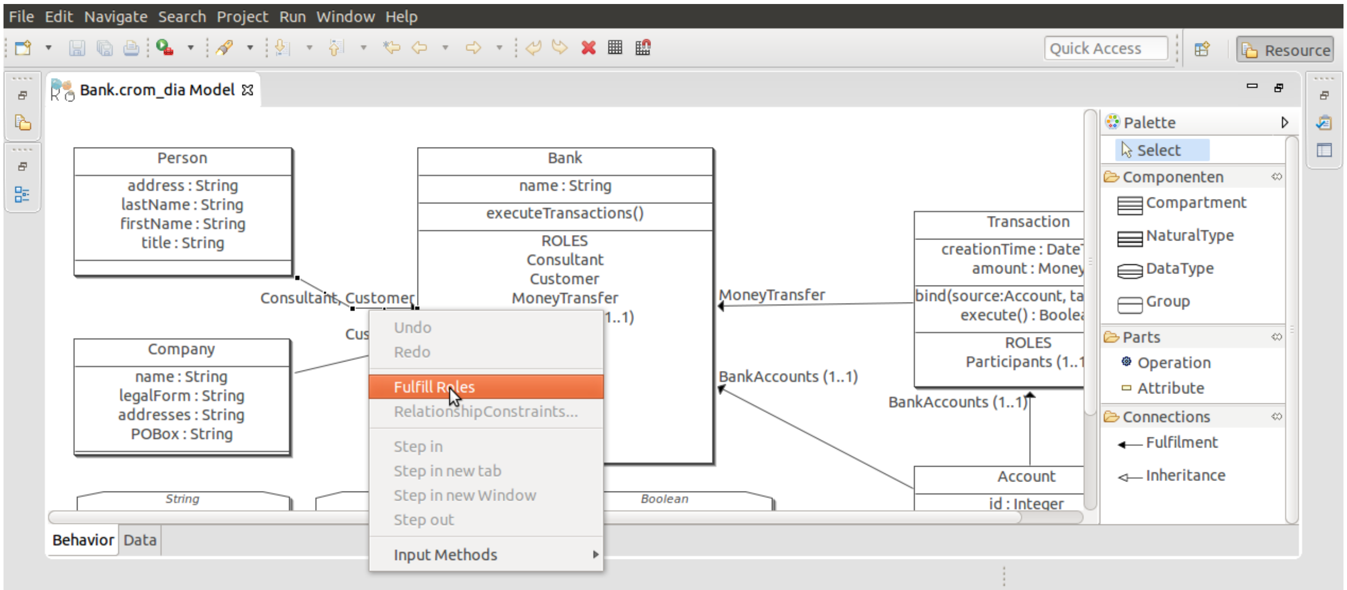

Figure 5: Refining the fulfillment relation

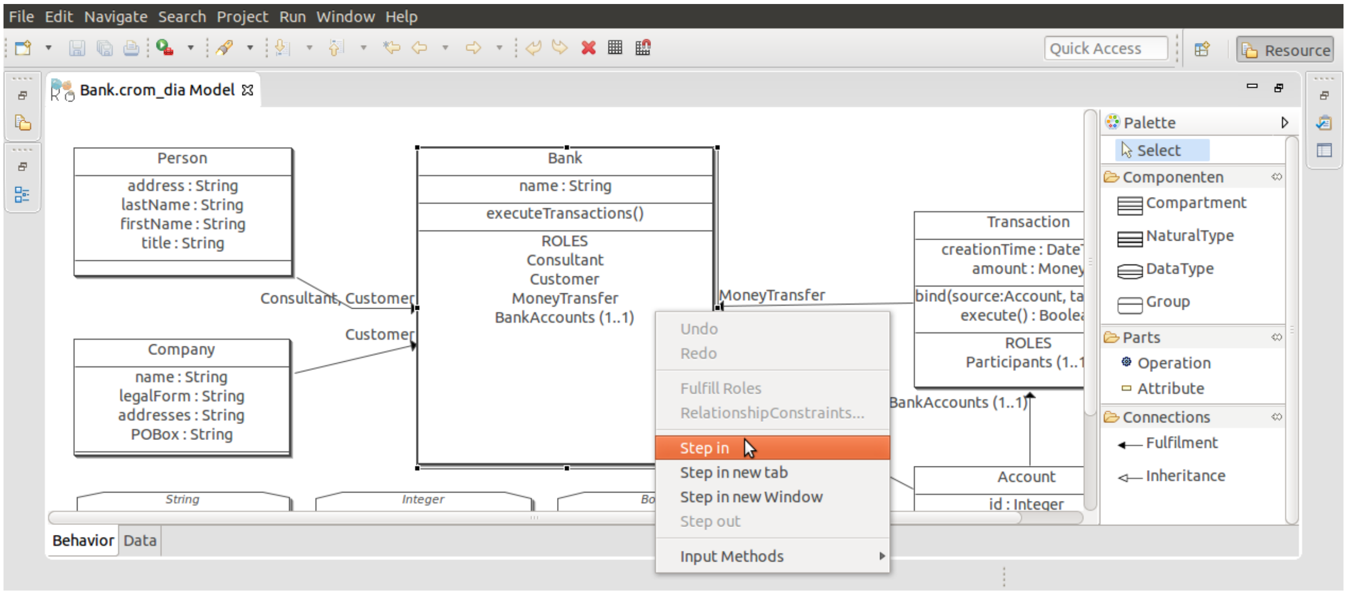

Figure 6: Stepping into compartment types

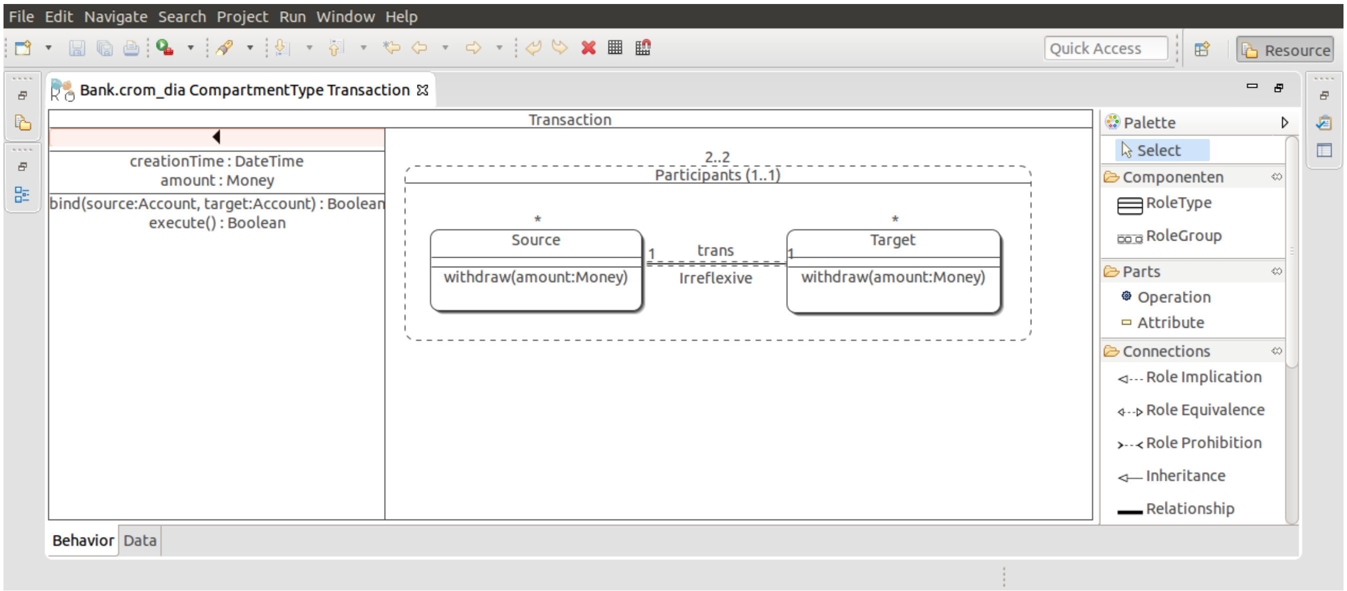

Figure 7: Focus view of the `transaction` compartment type

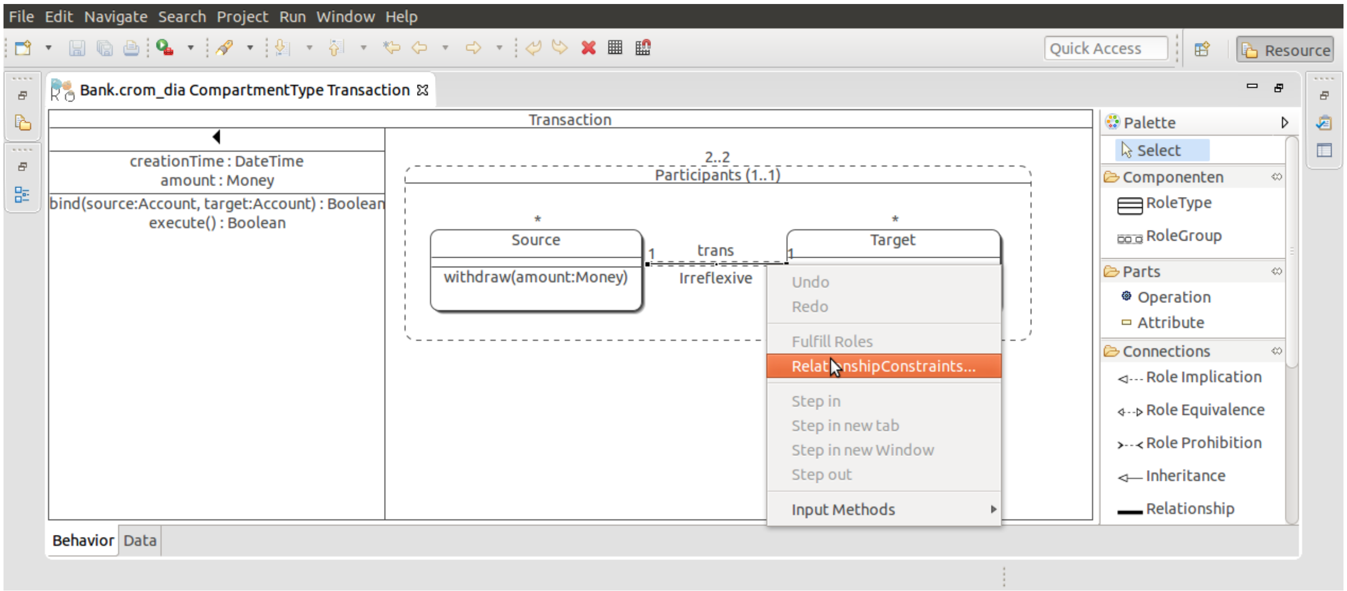

Figure 8: Creating relationship constraints

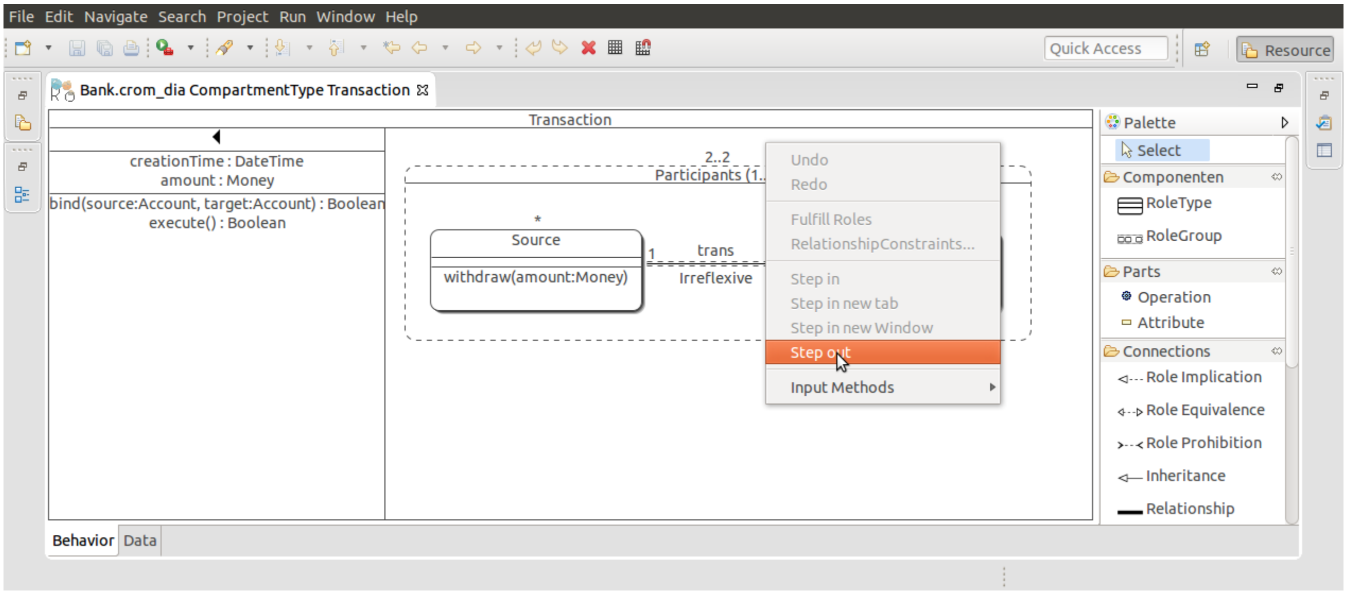

Figure 9: Stepping out of a compartment type