TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute of Software and Multimedia Technology, Software Technology Group

WS2017/18 – Model-driven Software Development in Technical Spaces

Role-oriented Modeling with FRaMED

Professor: Prof. Dr. Uwe Aßmann
Tutor: Dr.-Ing. Thomas Kühn

1 Role-based Modeling

Role-based Modeling is a well-researched modeling paradigm that is centered around the notion of objects playing multiple roles in multiple collaborations. In particular, more recent work introduced the notion of *Compartments*, to represent objectified collaborations establishing the context for a set of roles [1].

1.1 Task 1: The Compartment Role Object Model (CROM)

The *Compartment Role Object Model* (CROM) [1] is a novel role-based modeling language that fully embraces the context-dependent and relational nature of roles. Fig. 1 depicts a role-based model for *petri nets*. It contains two natural types **Entity** and **Node**. The former represent named model elements that can play the roles **Place** as well as **Transitions**. The latter can play the role **Required** and **Provided** representing arcs from a **Place** to a **Transition** and from a **Transition** to a **Place**, respectively.

In addition to that, CROM supports various model constraints. For this tutorial only three of them are interesting. First, *role groups*, depicted as dashed rounded rectangle with a cardinality in its head, limit the number of role types that can be played simultaneously by a particular object. Similarly, *role-prohibition* and *role-implication* [3] specify that two role types cannot be played together and one role type requires that the other is also played, respectively. Third, *occurrence constraints* are cardinalities placed above role types and role groups and impose a lower and upper bound to the number of instances of the corresponding role types. A detailed description of the modeling language can be found in [1].

1. Refine the presented role-based model (Fig. 1) by adding *occurrence constraints*, *role groups*, and *role constraints*. These should reflect the following constraints:
   - A **Node** cannot play a **Required** and a **Provided** role at the same time.
   - An **Entity** cannot play a **Place** and a **Transition** role at the same time.
   - A **PetriNet** should have at least a **Place** or a **Transition** to be valid.

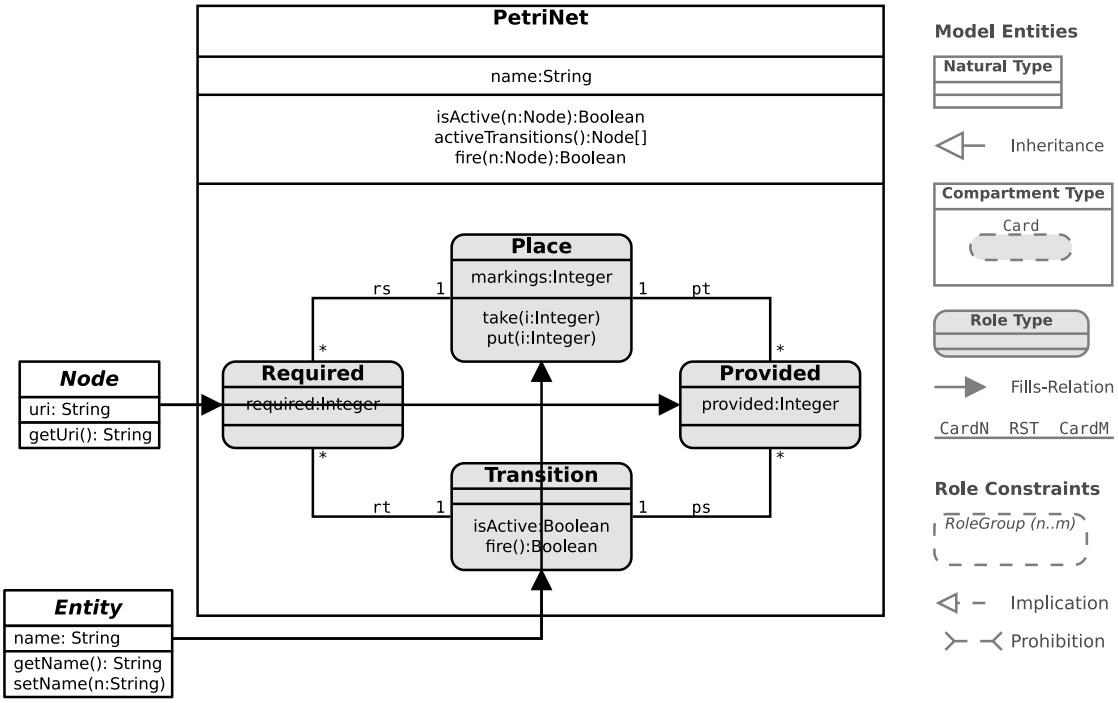

Figure 1: Compartment Role Object Model of petri nets

2. Enable the model to be represented in a Graphical Editor by adding a Compartment encapsulating the graphical representation of petri nets and link it to the corresponding natural types. Add the corresponding role types, relationship types, and fills relation to the incomplete model shown in (Fig. 2).
3. (Optional) Develop an extension to the petri net model (Fig. 1) that introduces *inhibitor arcs*. These connect a place with a transition, such that the transition can only fire if the place is empty. To do this, you should inherit the **PetriNet** compartment type when modeling an **InhibitorPetriNet**. This compartment automatically inherits all properties, role types, relationship types and constraints of its super type. Furthermore, it is possible to add properties, role types, relationships, and constraints or override existing properties of role types. Hence, you should add the necessary role types, relationships and constraints, as well as override properties of existing role types to model *inhibitor petri nets*.

1.2 Task 2: Full-fledged Role Modeling EDitor (FRaMED)

Although role-based modeling has a long tradition, there is almost no tool support for role-based modeling. To change this, the *First Role Modeling Editor* (FRaMED) has been developed to fully support all features of CROM and some more [2]. *FRaMED* is a Graphical Editor build with *GEF* as an Eclispe-plugin. It supports the creation and modification of *natural types* and *compartment types* in the top level view, as well

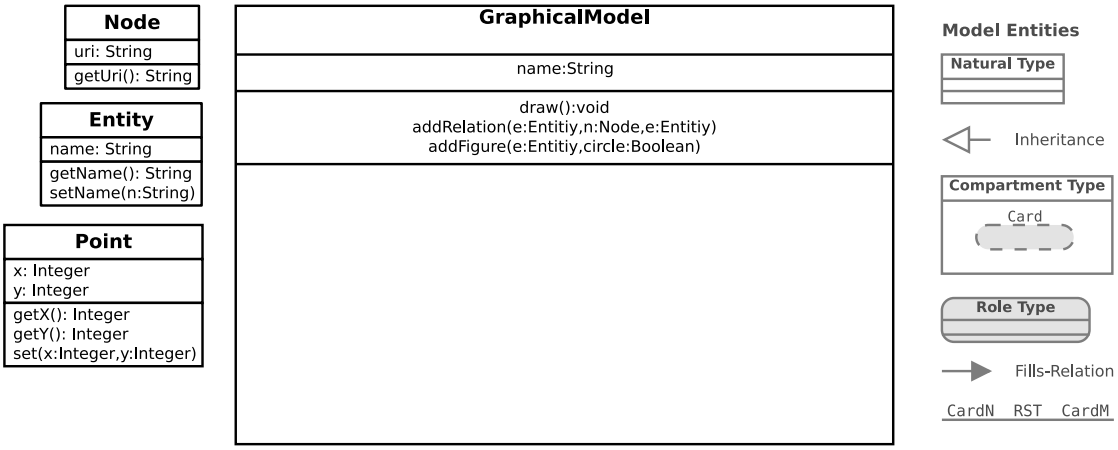

Figure 2: Compartment Role Object Model for a graphical model of petri nets

as the creation and modification of *role types* and *relationships* within a *compartment type* (click right on a *compartment type* and select *Step in*). Depending on the view, the palette shown on the right-hand side contains the available model elements and model relations.

1. Install and Run *FRaMED* following the step-by-step installation guide.[1]
2. Follow the tutorial in the *FRaMED wiki*[2] and model the *Bank Example*.
3. Design the **PetriNet** model shown above with *FRaMED*.
4. Extend **PetriNet** model in accordance to your solution of Task 1.
5. Extend **PetriNet** model in accordance to your solution of Task 2.
6. (Optional) File a *bug report* for all errors you have discovered.[3]

References

[1] Thomas Kühn, Böhme Stephan, Sebastian Götz, Christoph Seidl, and Uwe Aßmann. A combined formal model for relational context-dependent roles. In *Software Language Engineering*, pages 141–160. ACM, 2015.

[2] Thomas Kühn, Kay Bierzynski, Sebastian Richly, and Uwe Aßmann. FRaMED: Full-fledge role modeling editor (tool demo). In *Proceedings of the 2016 ACM SIGPLAN International Conference on Software Language Engineering*, SLE 2016, pages 132–136, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4447-0. doi: 10.1145/2997364.2997371. URL http://doi.acm.org/10.1145/2997364.2997371.

[3] Dirk Riehle and Thomas Gross. Role model based framework design and integration. In *ACM SIGPLAN Notices*, volume 33, pages 117–133. ACM, 1998.

[1] https://github.com/leondart/FRaMED/wiki/Step-by-Step-Installation
[2] https://github.com/leondart/FRaMED/wiki/Example
[3] https://github.com/leondart/FRaMED/issues