Name Consistency Checker in ATOM3

Presented by Victoria Yang
Overview

- Main constraint
- Constrain Checker (CC) revisited
- Implementation in ATOM3
- Demo
- Future Work
Goal

- Enforce name consistency over elements in a model.
- Petri Net Example.
Recall: The Constraint Checker (CC)

1. Map consistency constraint to checking rule:
   - *if Expression then Action*

2. Apply checking rule to model.
UML Well-formedness rule:

- If a contained element, which is not an Association or Generalization, has a name then the name must be unique in the Namespace.
Generalization of Rule

Well-formedness Rule:

- \textit{No instantiation of the same meta element may have the same name}
Petri Net Name Constraints

1. No two places may have the same name within a model.

2. No two transitions may have the same name within a model.
Map Name Constraint to Checking Rule

Checking rule:

- If *two instantiations of the same meta element have the same name* then *they should be the same instantiation.*
Problem with Checking Rule

- Checking rule implies should merge similarly name places and transitions into one.

- Problem:
  - if modeler intended for separate entities, merger will cause loss of info.

- Solution:
  - rename one entities instead of merging.
Alteration of Checking Rule

New Checking Rule:

- If *two instantiations of the same meta element has the same name* then *rename one of the instantiations*
Petri Net Checking Rules

1. If *two places have the same name within the model*, then *rename a place of them*

2. If *two transitions have the same name within the model*, then *rename a transition of them*
CC Implementation in ATOM3

- Implement constraint checking rules as graph grammars in ATOM3’s graph rewriting system.
Implementation Issues

Problems:

- Need to know whether entity has been checked.
- Need to trace changes made (which entities were renamed).
Solution: Use Stereotypes

- Extend metamodel with a tagged value.
The Tagged Values

- Implemented as an integer attribute of Place and Transition for Petri Net.

Tagged Values

- (isChecked, value)
- Value = 0: Initial
- Value = 1: Name is unique
- Value = 2: Name is not unique
Graph Rewriting Rules for CC

- Preprocessing

- Rule 1: Place Names
  - Add all names of places to the global name list
  - Set their tagged values to 1
Apply checking rules for places in CC

Rule 2: Unique Place Names
- Look up duplicated names in the global name list
- Rename non-unique names until no more duplicated names are found
- Add new name to global name list, remove its old name and set their tagged value to 2
Graph Rewriting Rules for CC (3)

- Rules for transition namespace
  - Similar to rules for place namespace
- Rule 3 : Preprocessing
- Rule 4 : Apply checking rules for transitions
Future Work with ATOM3

- Automation of extending meta element with tags
- Get user input to force action
References


