(In)Consistency Management

Arend Rensink and Eugene Syriani

Context

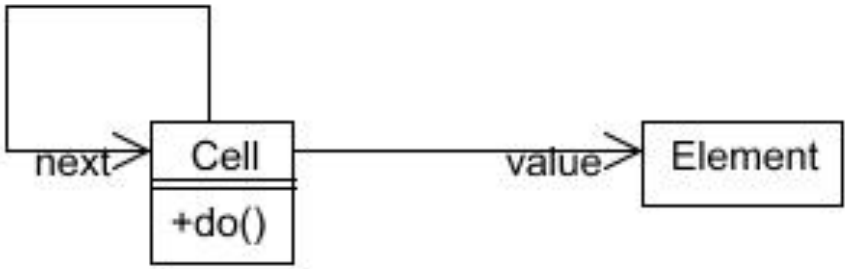
- Type of inconsistency under study:
  - Horizontal
  - Intermodel
- To study inconsistency, first define consistency
- Consistency definition:

  Given 2 languages, a consistency requirement is a dependency relation between elements of the meta-models

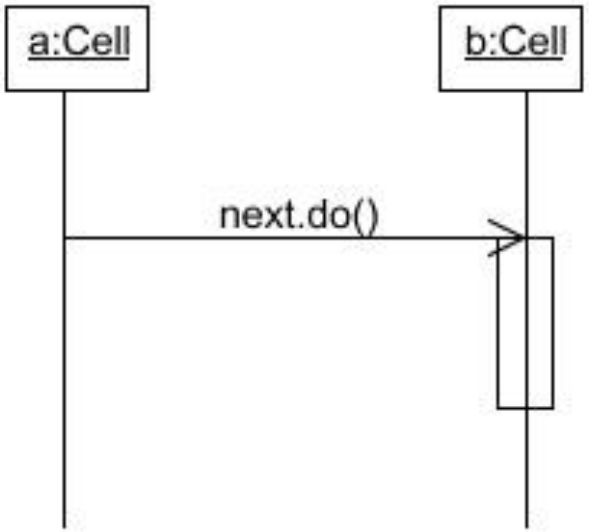
Context

- We concentrate on:
  - Models that do not have the same purpose
  - They model different aspects of a same system
  - Inconsistency may only occur on concepts for which they overlap

Example Models

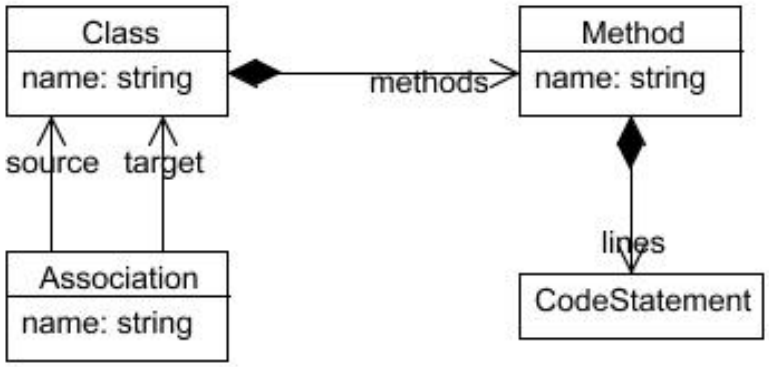


"Class Diagram"

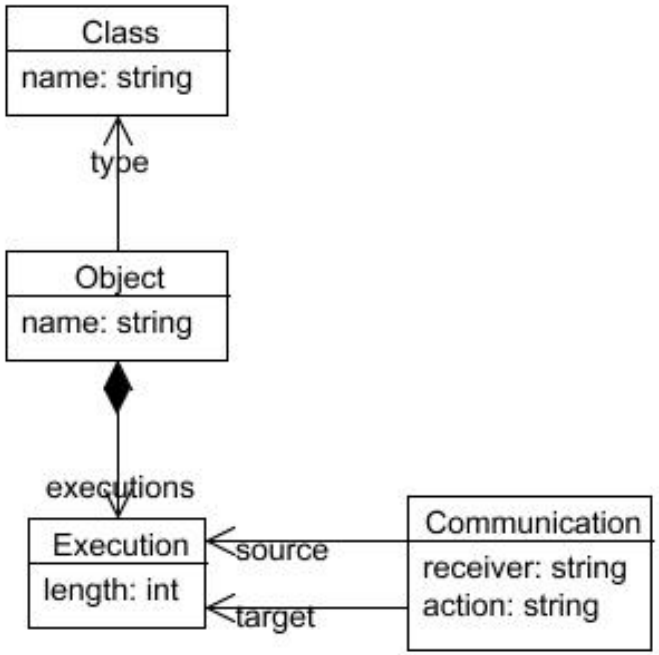


"Sequence Diagram"

Example Meta-Models



"Class Diagram"

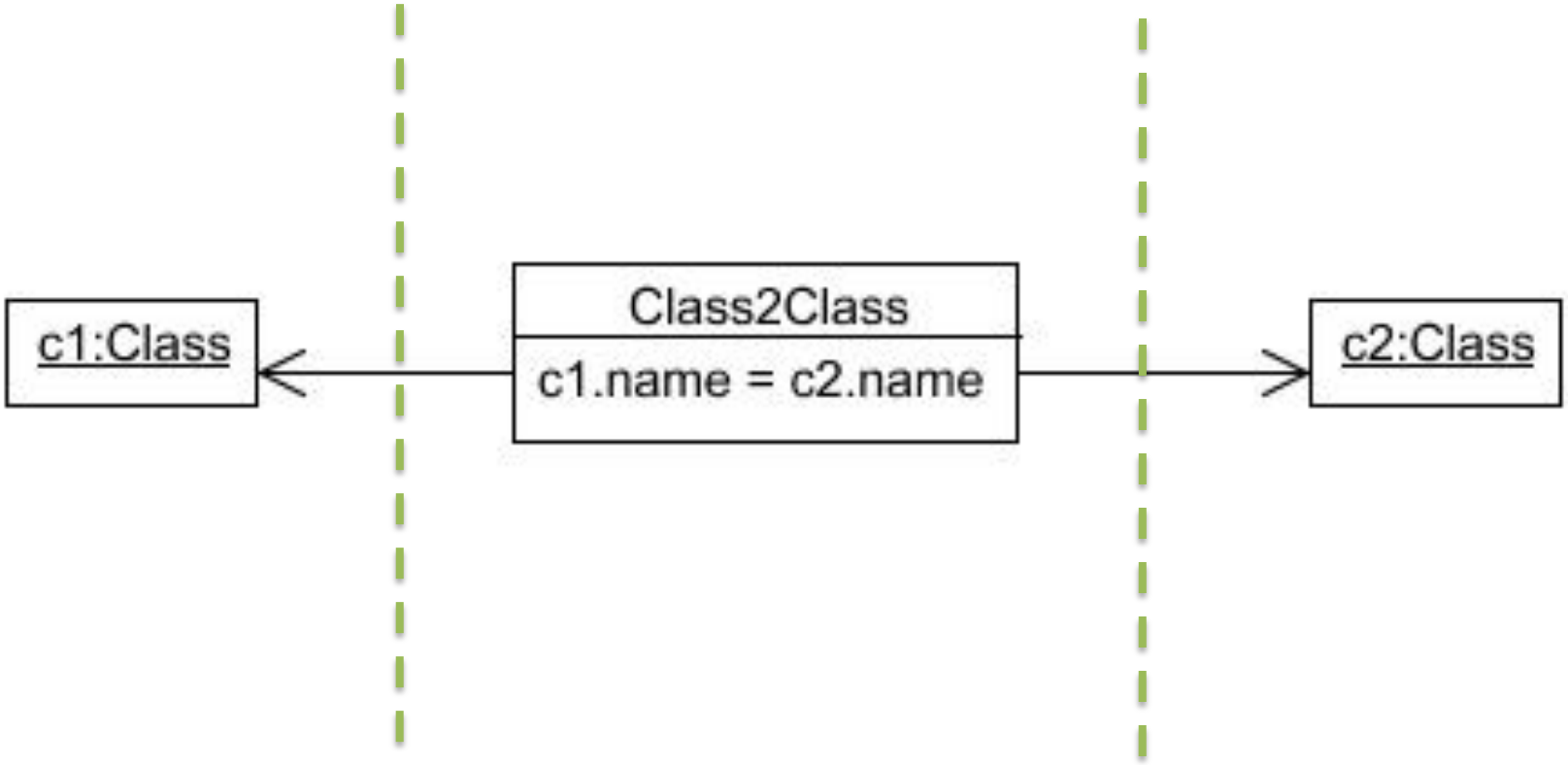


"Sequence Diagram"

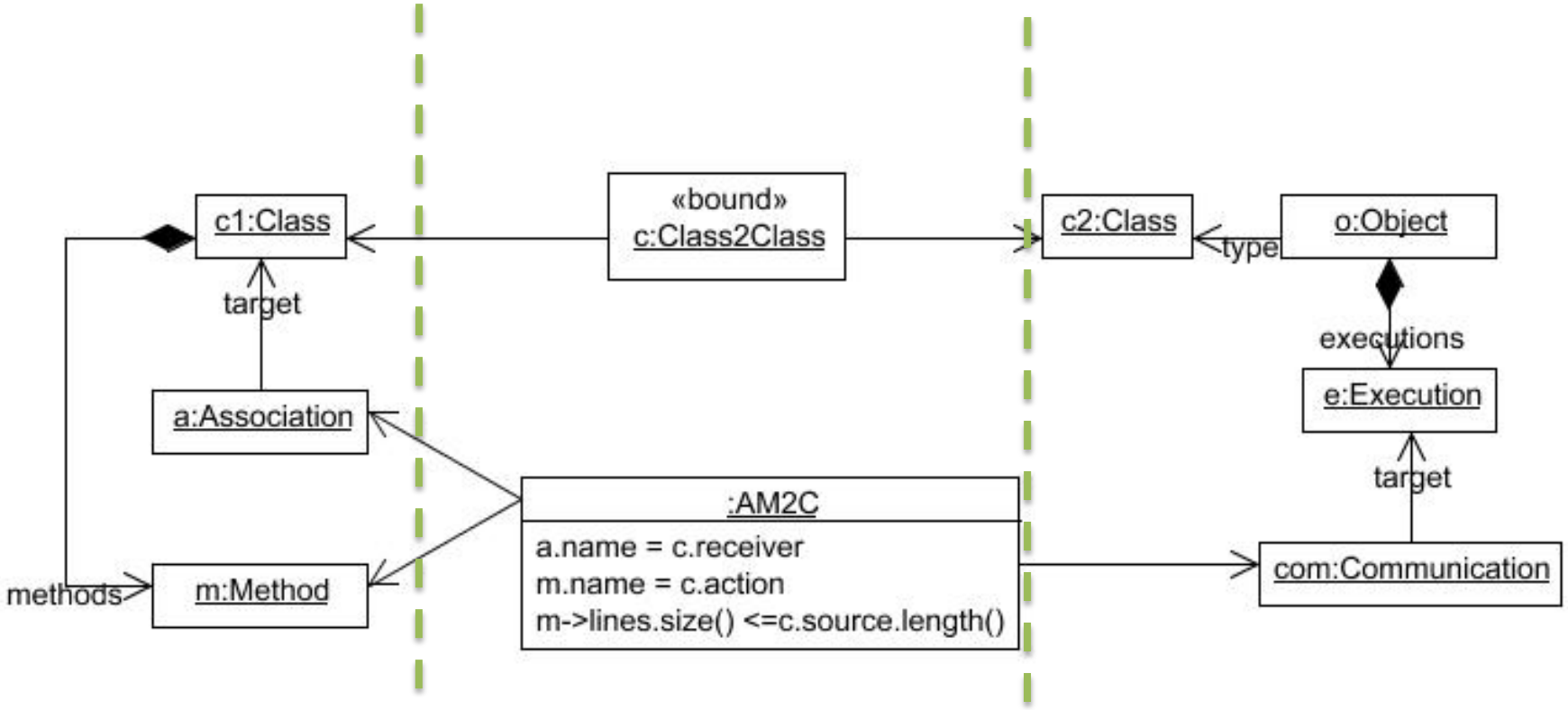
Syntax for Consistency Relation

- Use of a TGG-like notation to express consistency relation
- However, our purpose is NOT model transformation, but a relation definition

Class Consistency Relation



Communication Consistency Relation



$\forall com : \exists c,a,m$: pattern holds

Typical Scenario for Inconsistency

- Models are created or evolve
  1. After creation of a model
     - Run dependency check
     - Add correspondences automatically, manually or both
     - In case of inconsistency, flag them
  2. After a change of one of the models
     - Assuming we start with 2 consistent models (correspondence model has been created)
     - Structured editing operation
     - Keep history of edit events
       - What do we keep track of? (undo info / complete model /…)
     - Suggest repair actions

Repair Actions

- Repair in non-edited model
  - Use event history to locate violation point
  - In some cases, propose repairs
  - Flag inconsistencies

  Possible: rename class, method

  delete association -> delete communication

  Hard: method body longer
- Repair in edited model
  - ?Combination of the history events together with previously user-defined repair mechanisms?