Role-Playing Game Modeling in VMTS

Dylan Kiss

University of Antwerp
dylan.kiss@student.uantwerpen.be

Overview

- About VMTS
- The RPG formalism
- Abstract syntax
- Operational semantics
- VMTS vs. AToMPM
- Conclusion

Overview

- About VMTS
- The RPG formalism
- Abstract syntax
- Operational semantics
- VMTS vs. AToMPM
- Conclusion

About VMTS

- Graph-based metamodeling and model processing framework
- N-layer
- Budapest University of Technology and Economics
- Allows you to define, customize and utilize
  - Languages
  - Transformations

T. Levendovszky, L. Lengyel, G. Mezei, H. Charaf, A Systematic Approach to Metamodeling Environments and Model Transformation Systems in VMTS, Electronic Notes in Theoretical Computer Science 127 (1) (2005) 65-75

Overview

- About VMTS
- The RPG formalism
- Abstract syntax
- Operational semantics
- VMTS vs. AToMPM
- Conclusion

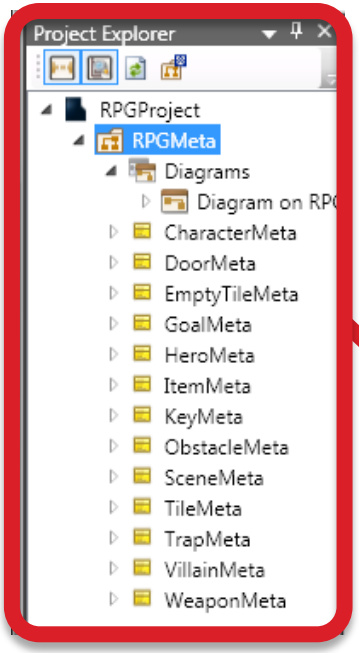
The RPG formalism

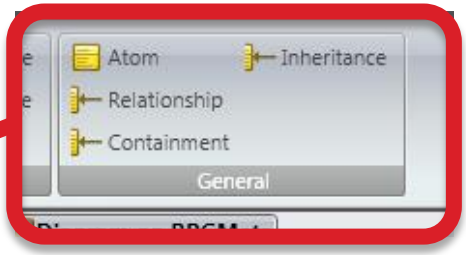
- One or more scenes
- Each scene consists of connected tiles
  - EmptyTile, Obstacle, Trap or Door
- Two types of characters
  - Hero (exactly one) and Villain
- Three types of items
  - Goal (at least one), Weapon and Key
- Non-obstacle tile can contain one character and item

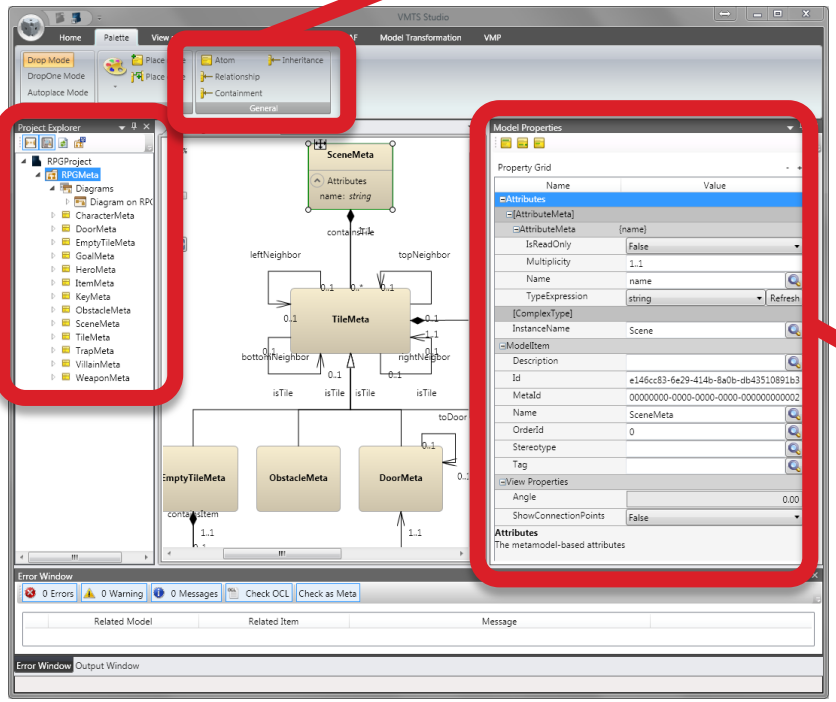
Overview

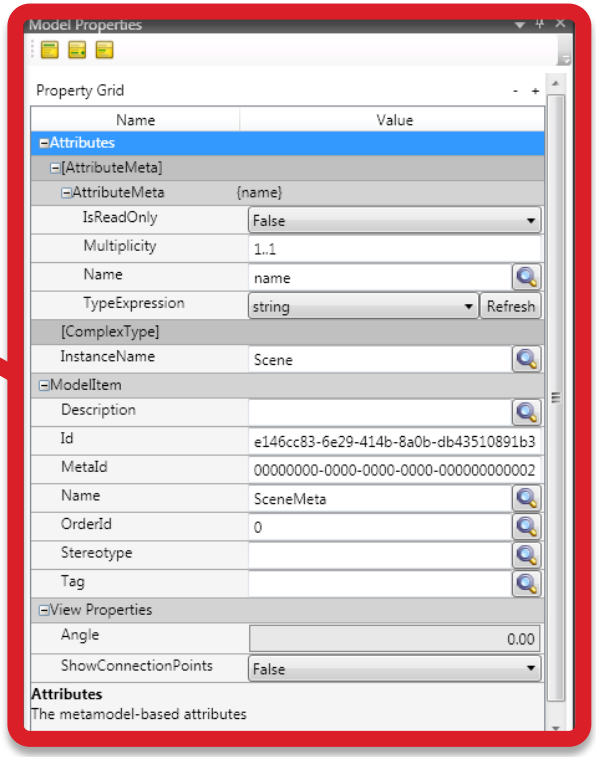
- About VMTS
- The RPG formalism
- Abstract syntax
- Operational semantics
- VMTS vs. AToMPM
- Conclusion

Abstract syntax

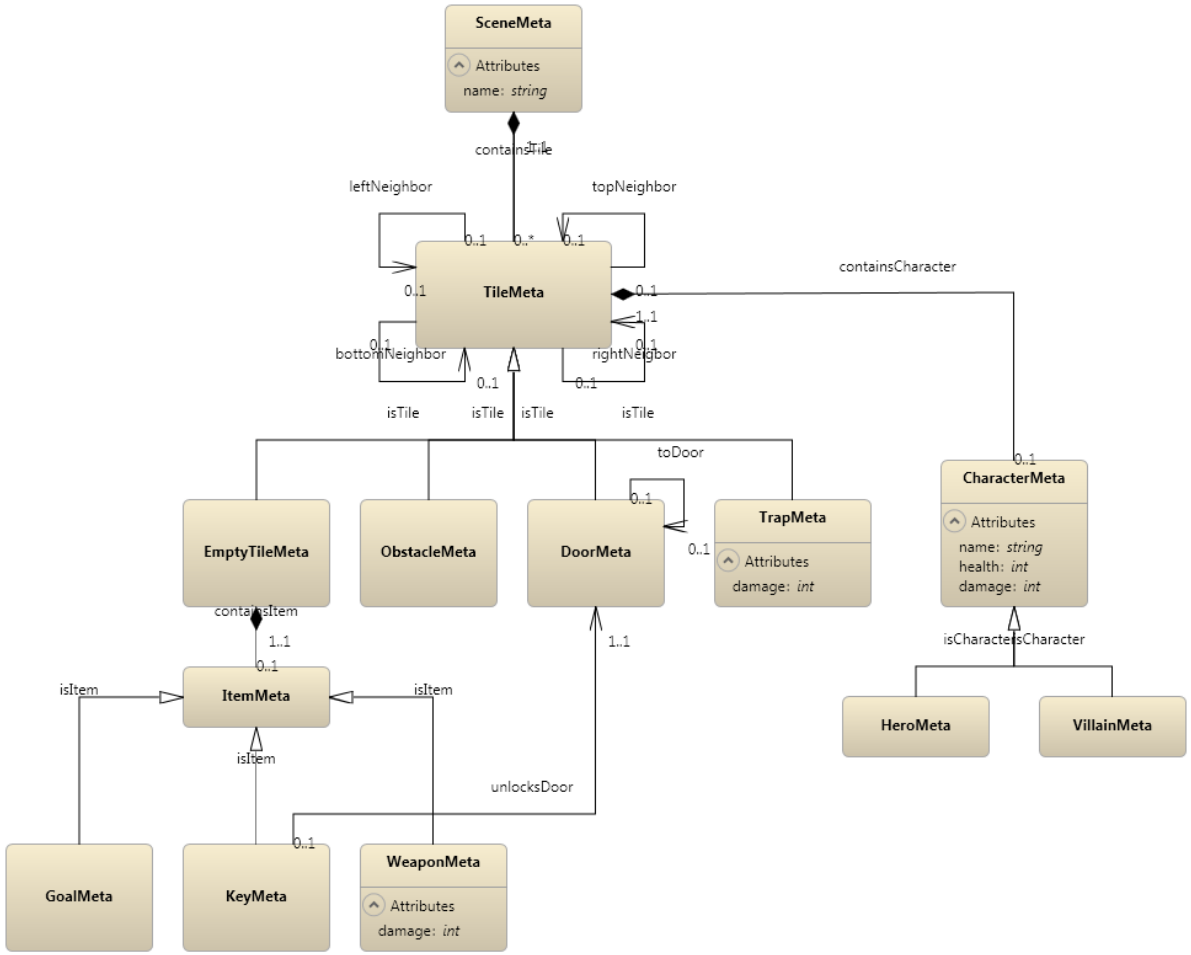




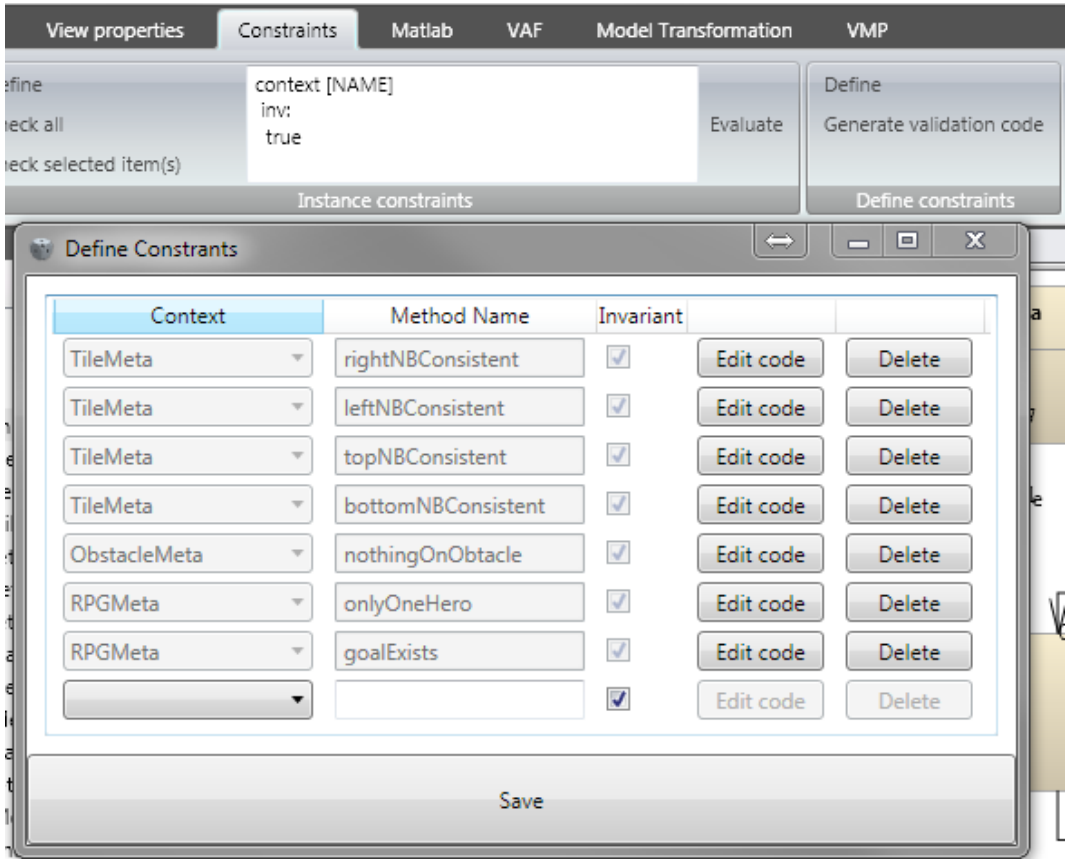




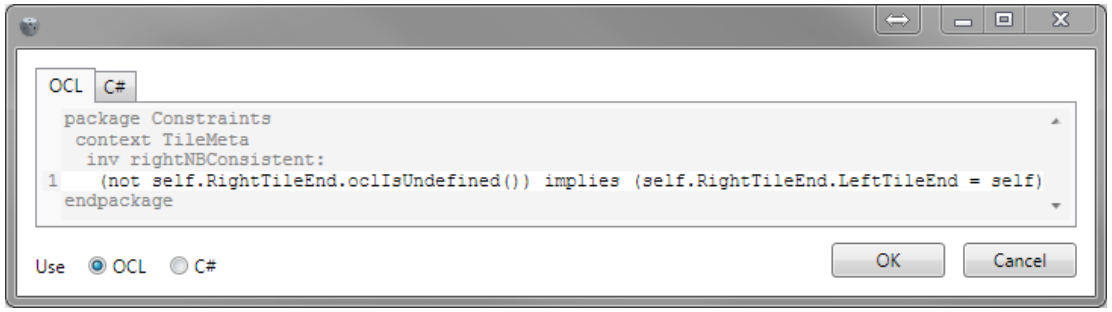
Abstract syntax



Abstract syntax: constraints



Abstract syntax: constraints




```
package Constraints
  context TileMeta
    inv rightNBConsistent:
      (not self.RightTileEnd.oclIsUndefined())
      implies (self.RightTileEnd.LeftTileEnd = self)
endpackage
```

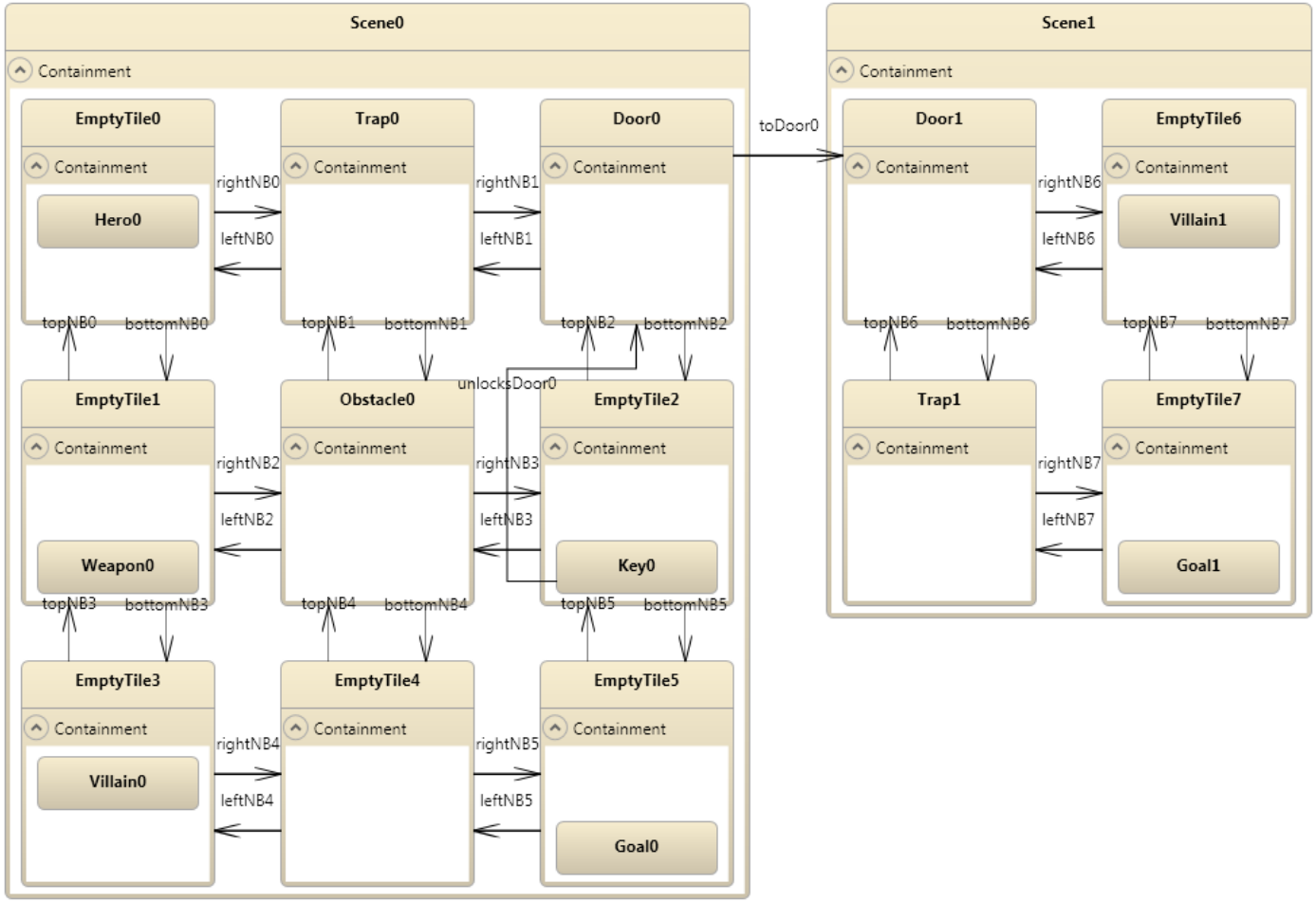
Abstract syntax: constraints

```
package Constraints
  context ObstacleMeta
    inv nothingOnObstacle:
      self.Character.oclIsUndefined()
endpackage
```

```
package Constraints
  context RPGMeta
    inv onlyOneHero:
      self.HeroMeta->size() = 1
endpackage
```

```
package Constraints
  context RPGMeta
    inv goalExists:
      self.GoalMeta->size() > 0
endpackage
```

Abstract syntax: example model



Overview

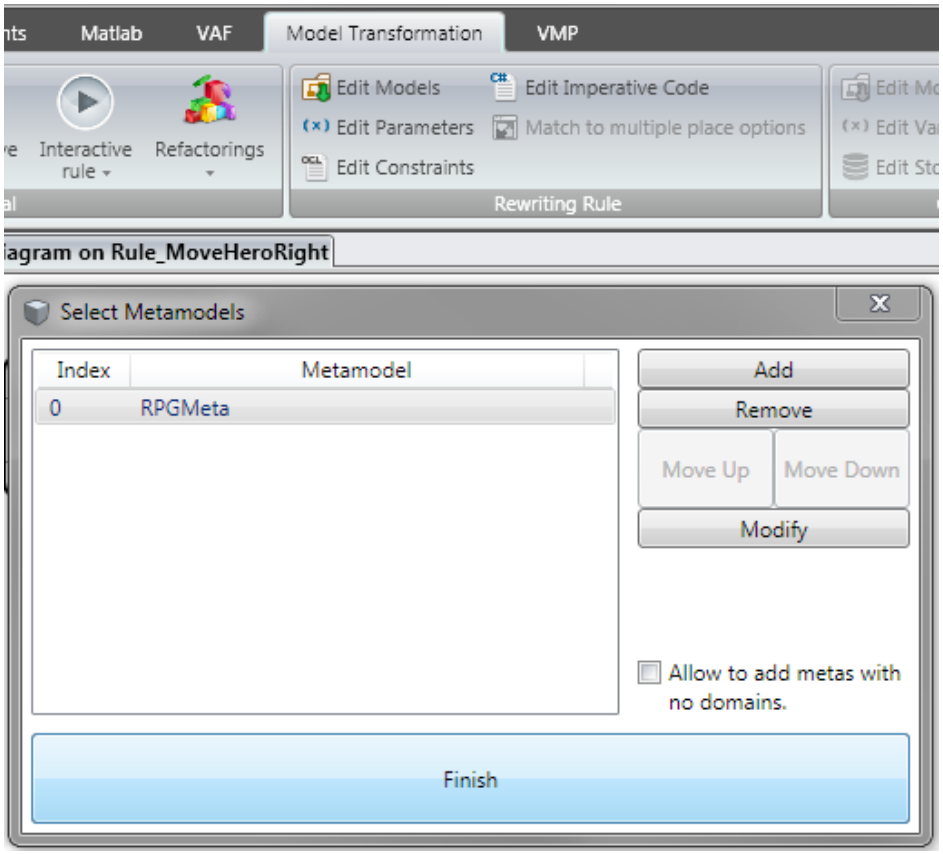
- About VMTS
- The RPG formalism
- Abstract syntax
- Operational semantics
- VMTS vs. AToMPM
- Conclusion

Operational semantics

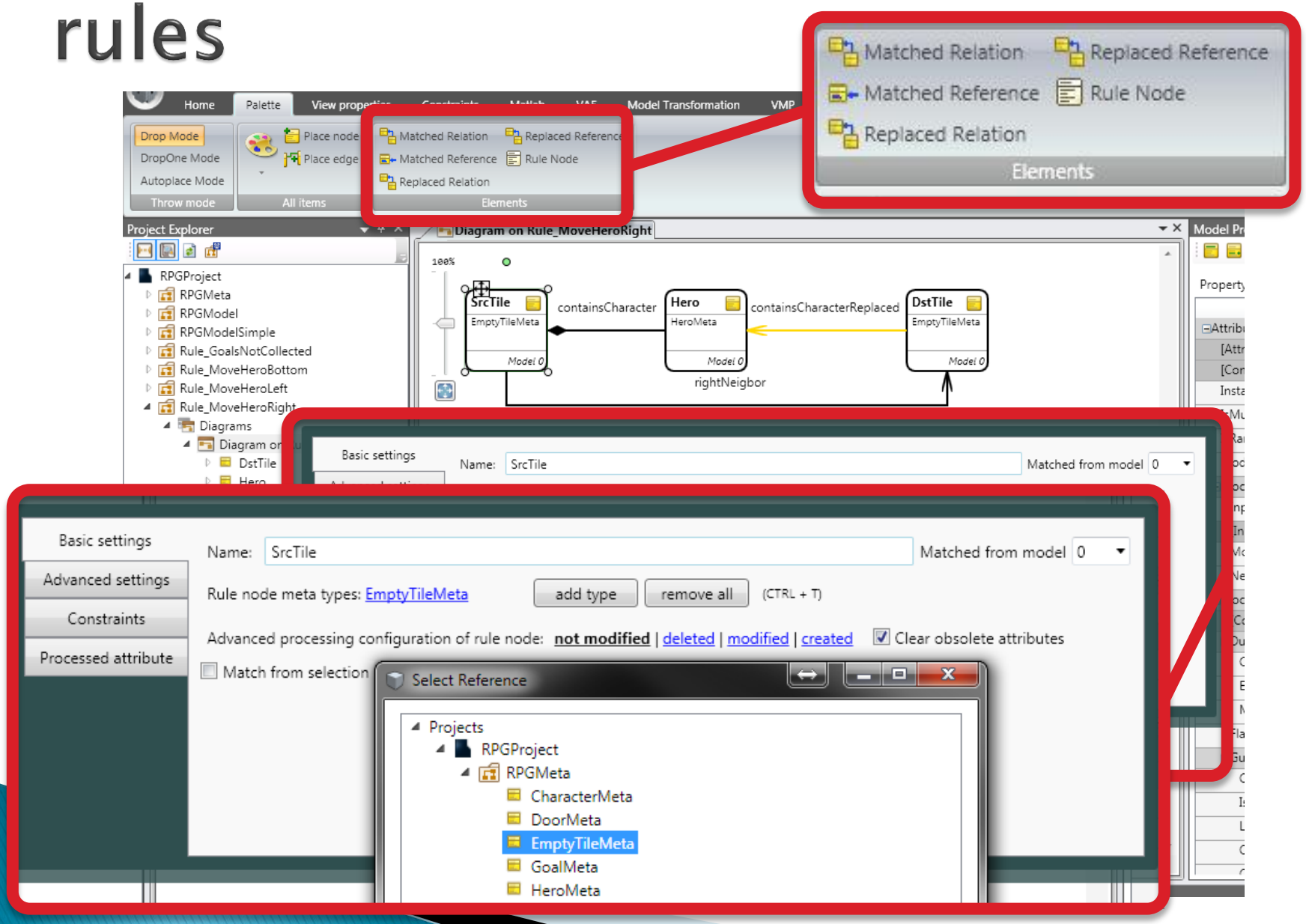
- Based on graph rewriting
- Two steps
  - Creating rewrite rules
  - Creating a transformation control flow

Operational semantics: rewrite rules

- New model of type MTRMETA
  - First: metamodel(s)



Operational semantics: rewrite rules

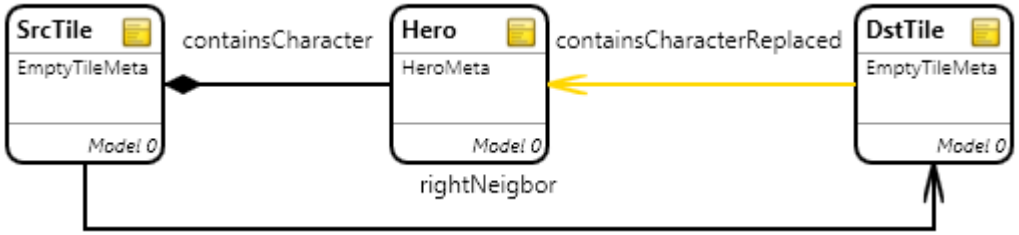


Operational semantics: rewrite rules

- Compiling rule constraints and actions failed
- Restrictions:
  - There are **no villains** in the game
  - There are only **two types of tiles** in the game: **EmptyTile** and **Obstacle**
  - There is only **one type of item** in the game: **Goal**
- 6 rewrite rules:
  - MoveHeroLeft/Right/Top/Bottom
  - PickUpGoal
  - GoalsNotCollected

Operational semantics: rewrite rules

MoveHeroRight

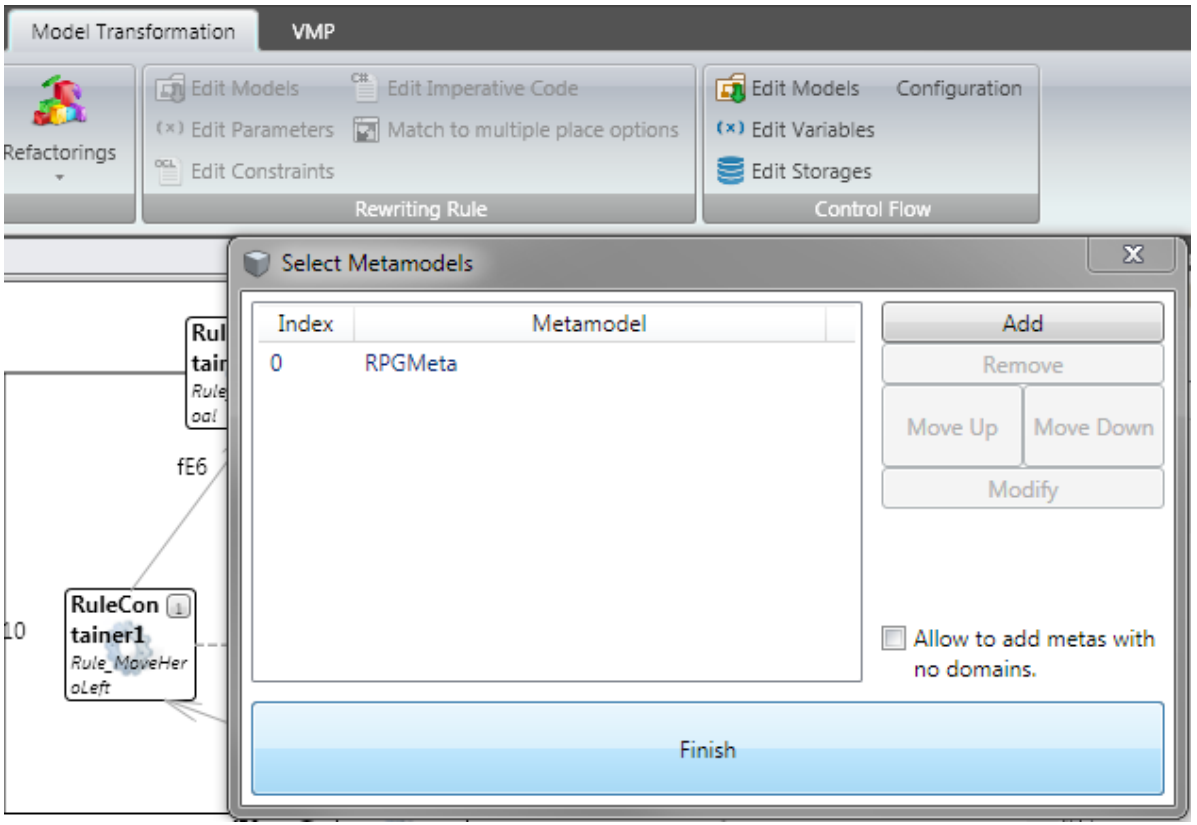


PickUpGoal

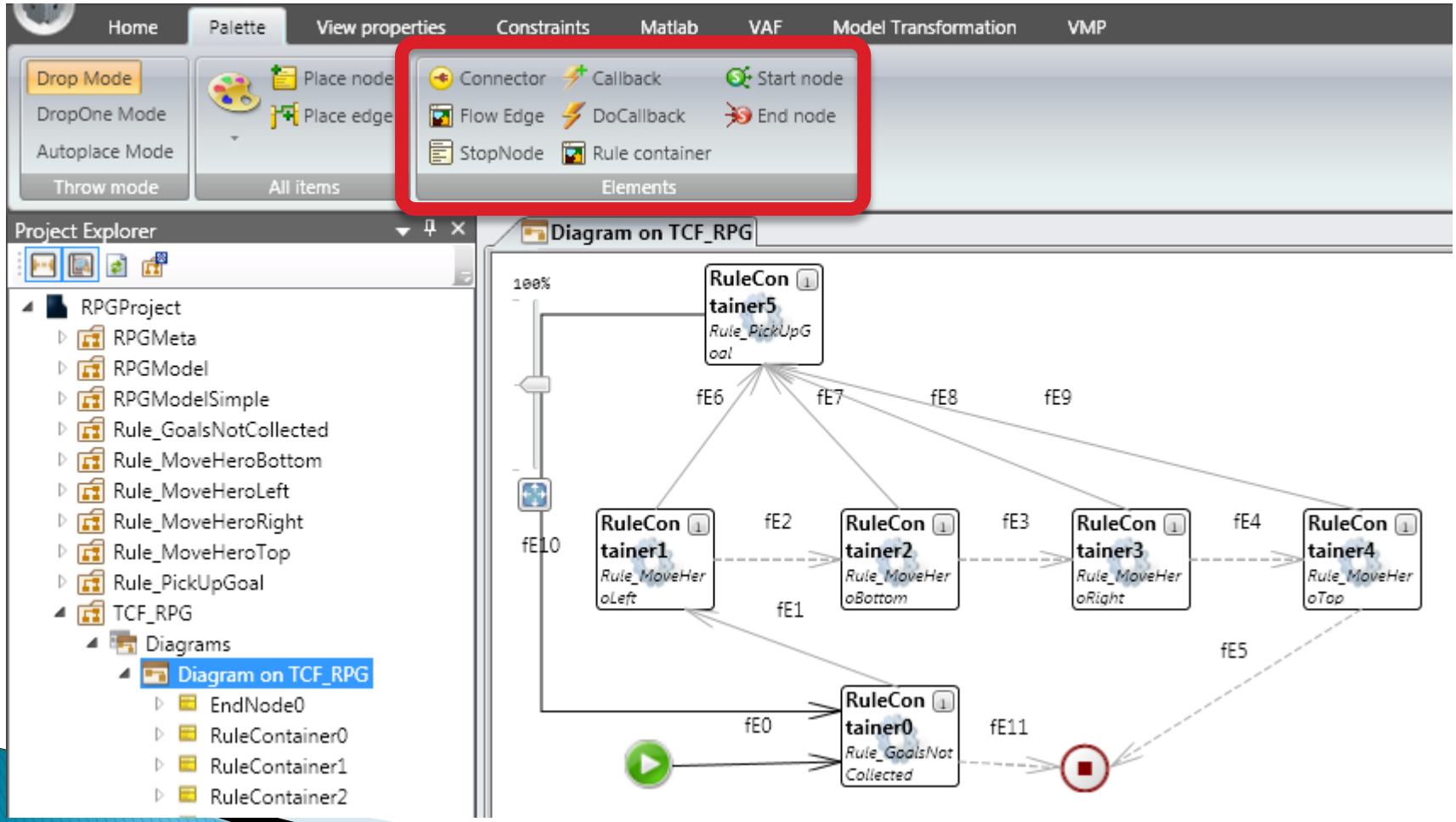
GoalsNotCollected

Operational semantics: transformation control flow

- New model of type TCFMETA
  - First: metamodel(s)

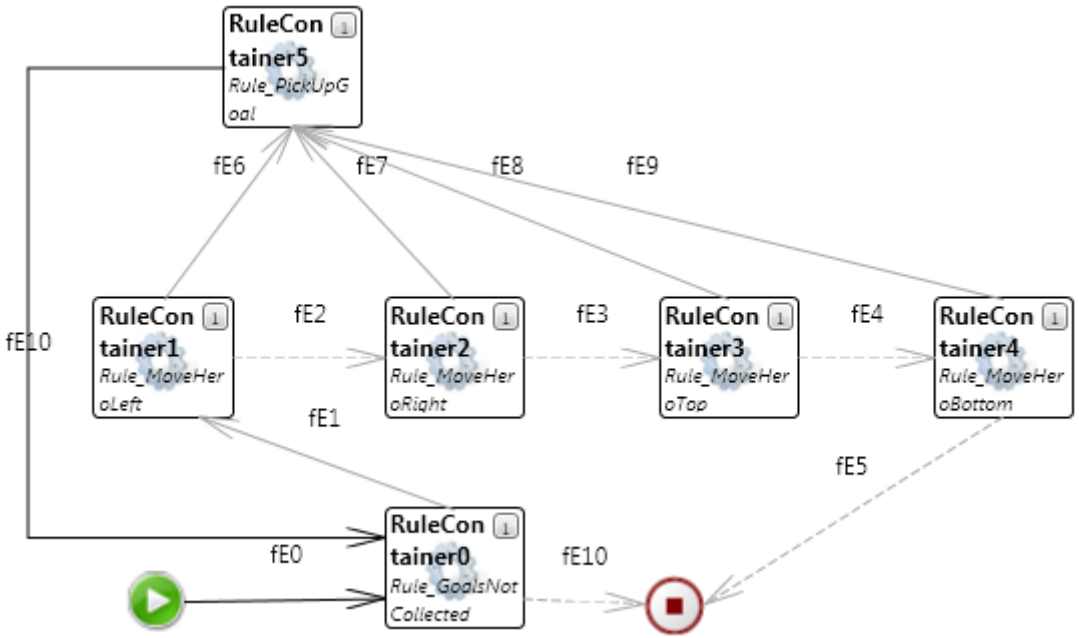


Operational semantics: transformation control flow



Operational semantics: transformation control flow

- No support for randomness
  - Deterministic order of rules
  - Unable to properly simulate an RPG



Overview

- About VMTS
- The RPG formalism
- Abstract syntax
- Operational semantics
- VMTS vs. AToMPM
- Conclusion

VMTS vs. AToMPM

- Abstract syntax
  - Very similar in both tools
  - Simplified class diagram
  - Constraints

  - **VMTS**: n-layer

VMTS vs. AToMPM

- Concrete visual syntax
  - Major difference
  - **AToMPM**: built-in visual syntax creation
  - **VMTS**: create your own plugin for visualization using WPF (Windows Presentation Foundation)

VMTS vs. AToMPM

- Operational and denotational semantics
  - Both use graph rewriting
  - Visual representation differs
  - **AToMPM**: clearly divided LHS, RHS, NAC
  - **VMTS**: one graph, no NAC

VMTS vs. AToMPM

- Modeling environment
  - **AToMPM**: cross-platform, web-based
  - **VMTS**: written in C# using .NET framework, only Windows platform

Overview

- About VMTS
- The RPG formalism
- Abstract syntax
- Operational semantics
- VMTS vs. AToMPM
- Conclusion

Conclusion

- Intentional plan
  - Abstract syntax
  - Concrete visual syntax
  - Operational semantics
  - Denotational semantics

Conclusion

- Intentional plan
  - Abstract syntax
  - Concrete visual syntax
    - requires developing plugin with WPF
  - Operational semantics
    - partly possible: constraints/actions/NAC/randomness
  - Denotational semantics
    - Impossible: constraints/NAC

Conclusion

- VMTS
  - + n-layer
  - + default visual syntax
- AToMPM
  - + web-based tool
  - + built-in visual syntax editor
  - + randomness and NAC

- My experience: AToMPM more user-friendly

Questions?