A Textual Syntax with Tool Support for the Goal-oriented Requirement Language

Vahdat Abdelzad, Daniel Amyot, Sanaa A. Alwidian, and Timothy Lethbridge

University of Ottawa, Ottawa, Canada

email: v.abdelzad@uottawa.ca

Outline

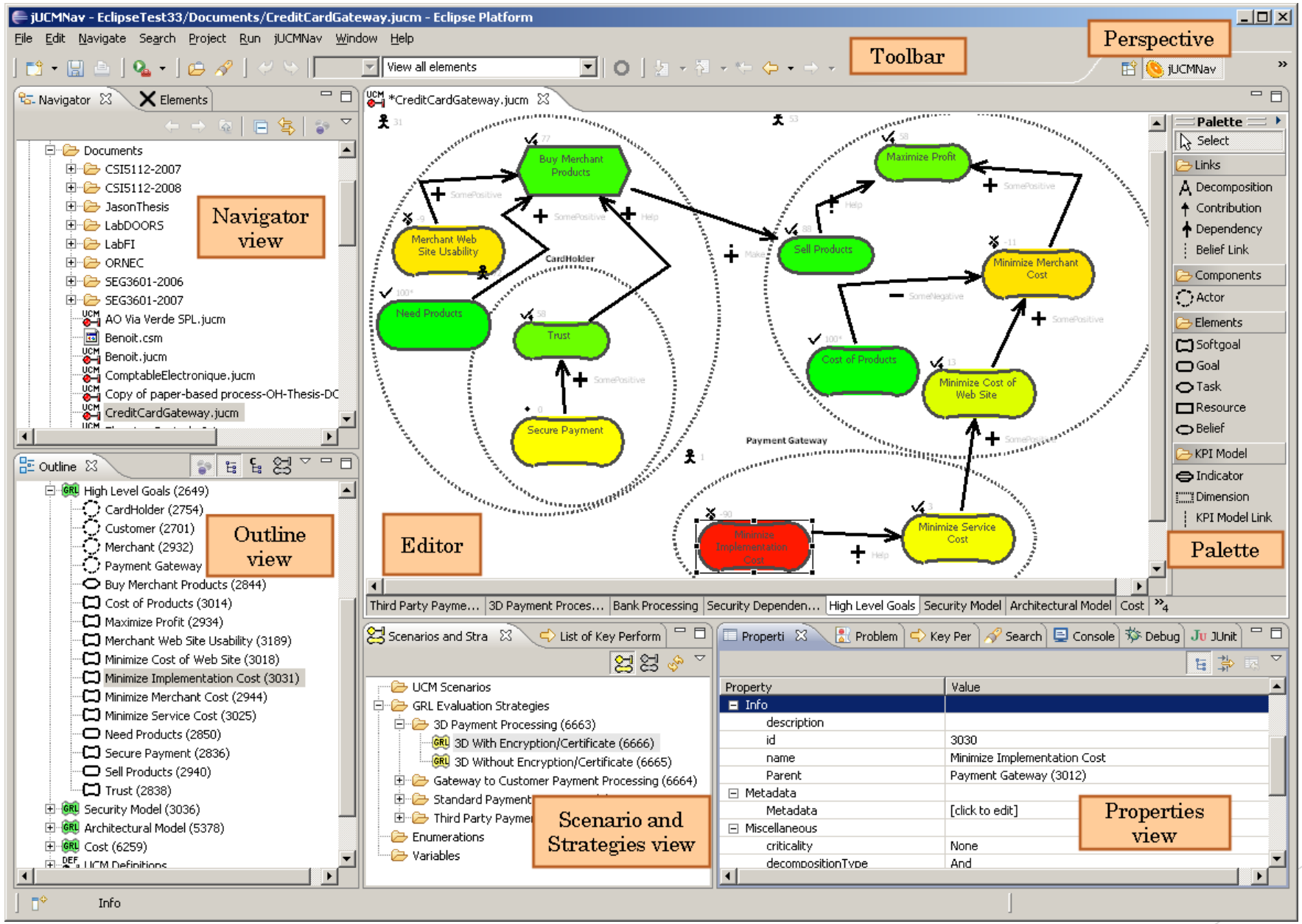
- Goal-oriented Requirement Language
- The Current Challenges
- Other Textual Syntax for Goal Modeling
- TGRL: A Textual Syntax for GRL
- TGRL Editor and Transformation
- Conclusion and Future Work

Goal-oriented Requirement Language (GRL)

- **Models the intentions of actors and systems, together with their various relationships**
- **GRL core concepts:**
  - Actors
  - Intentional elements (e.g., goals, softgoals, tasks, resources and beliefs)
  - *links* (decompositions, dependencies, weighted contributions)
  - Indicators

Goal-oriented Requirement Language (GRL) cont.

- **Metadata and typed URN links**
- **GRL model analysis through strategies**
  - Qualitative (using contribution, satisfaction, and importance values from their respective enumerated types)
  - Quantitative (using integer values in specific ranges)
- **Contribution changes**
- **Supported by jUCMNav**


jUCMNav - EclipseTest33/Documents/CreditCardGateway.jucm - Eclipse Platform
File Edit Navigate Search Project Run jUCMNav Window Help
Toolbar
Perspective
Navigator
Elements
Documents
CSI5112-2007
CSI5112-2008
JasonThesis
LabDOORS
LabFI
ORNEC
SEG3601-2006
SEG3601-2007
AO Via Verde SPL.jucm
Benoit.csm
Benoit.jucm
ComptableElectronique.jucm
Copy of paper-based process-OH-Thesis-DC
CreditCardGateway.jucm
Navigator view
Outline
High Level Goals (2649)
CardHolder (2754)
Customer (2701)
Merchant (2932)
Payment Gateway
Buy Merchant Products (2844)
Cost of Products (3014)
Maximize Profit (2934)
Merchant Web Site Usability (3189)
Minimize Cost of Web Site (3018)
Minimize Implementation Cost (3031)
Minimize Merchant Cost (2944)
Minimize Service Cost (3025)
Need Products (2850)
Secure Payment (2836)
Sell Products (2940)
Trust (2838)
Security Model (3036)
Architectural Model (5378)
Cost (6259)
Outline view
*CreditCardGateway.jucm
Buy Merchant Products
Merchant Web Site Usability
CardHolder
Need Products
Trust
Secure Payment
Maximize Profit
Sell Products
Minimize Merchant Cost
Cost of Products
Minimize Cost of Web Site
Payment Gateway
Minimize Implementation Cost
Minimize Service Cost
Editor
Palette
Select
Links
Decomposition
Contribution
Dependency
Belief Link
Components
Actor
Elements
Softgoal
Goal
Task
Resource
Belief
KPI Model
Indicator
Dimension
KPI Model Link
Palette
Third Party Payme...
3D Payment Proces...
Bank Processing
Security Dependen...
High Level Goals
Security Model
Architectural Model
Cost
Scenarios and Stra
List of Key Perform
UCM Scenarios
GRL Evaluation Strategies
3D Payment Processing (6663)
3D With Encryption/Certificate (6666)
3D Without Encryption/Certificate (6665)
Gateway to Customer Payment Processing (6664)
Standard Payment
Third Party Payme
Enumerations
Variables
Scenario and Strategies view
Properti
Problem
Key Per
Search
Console
Debug
JUnit
Property
Value
Info
description
id
3030
name
Minimize Implementation Cost
Parent
Payment Gateway (3012)
Metadata
Metadata
[click to edit]
Miscellaneous
criticality
None
decompositionType
And
Properties view
Info

What's the Issue?

- Creating/modifying goal models is often a tedious task with current graphical environments.
- It's not easy to create GRL models with complex features (e.g., strategies and contribution overrides)
- It is difficu… offers good… …nd purposes.
- We don't know how much graphical syntax is good for the modeling

explore the design of a textual syntax for GRL

Other Textual Syntax for Goal Modeling

- Liu and Yu [1] provided a textual grammar and an XML-based interchange format
  - Not simple and easy to be read by human (less cognitively affective)
- Formal Tropos‘s textual syntax [2]
  - supports an inner layer for declaring constraints on attributes and supports temporal logic properties
  - is more declarative, verbose, and limited in scope
  - does not have a feature-rich editor

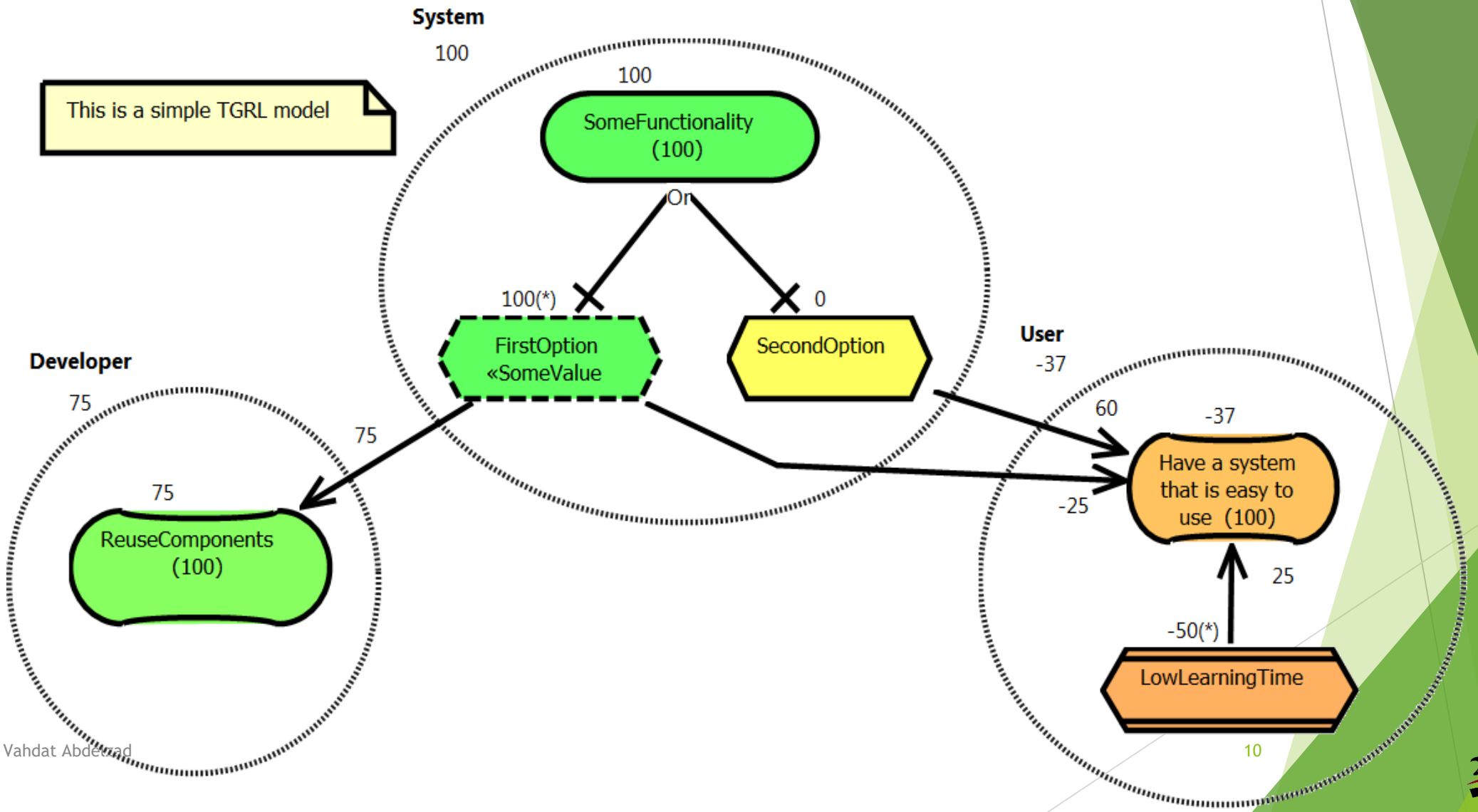
TGRL: A Textual Syntax for GRL

- **Used guiding principle inspired from the design of Umple**
  - Simplicity, consistency, and a programming language-like look and feel
- **Aligned the syntax and especially keywords with jUCMNav's metamodel**

TGRL: General Rules

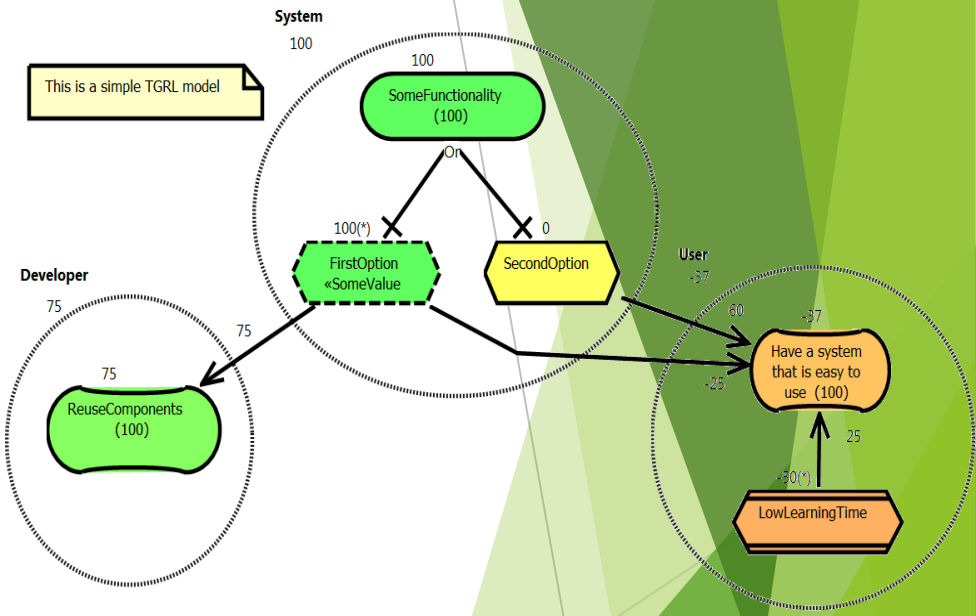
- GRL elements are usually defined through keywords using CamelCase boundaries (e.g., a softgoal intentional element is represented by a softGoal).
- String values are surrounded by quotation marks.
- Model element properties and sub-elements (if any) are set inside curly brackets.
- Every definition ends with a semicolon except when a pair of curly brackets is utilized to include either sub-elements or properties.
- Comments are delimited by // and /*…*/
- Most elements have a textual identifier (ID) as well as optional metadata (name-value pairs).

A Simple Example



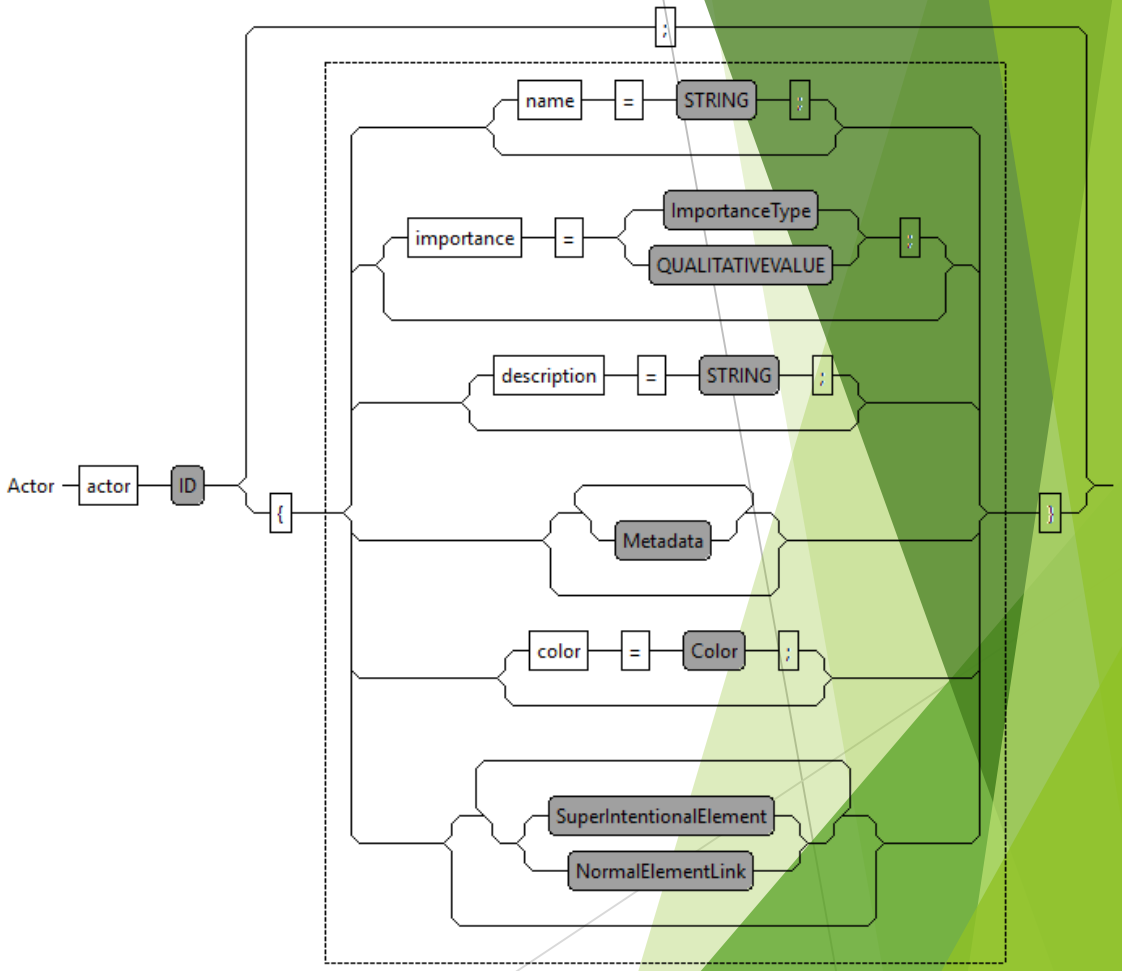
Main structure and Comments

```
grl IStar2015 {

    // A Graphical Model comment
    comment "This is a simple TGRL model";

    /*
     * textual modeling comment
     */

}
```

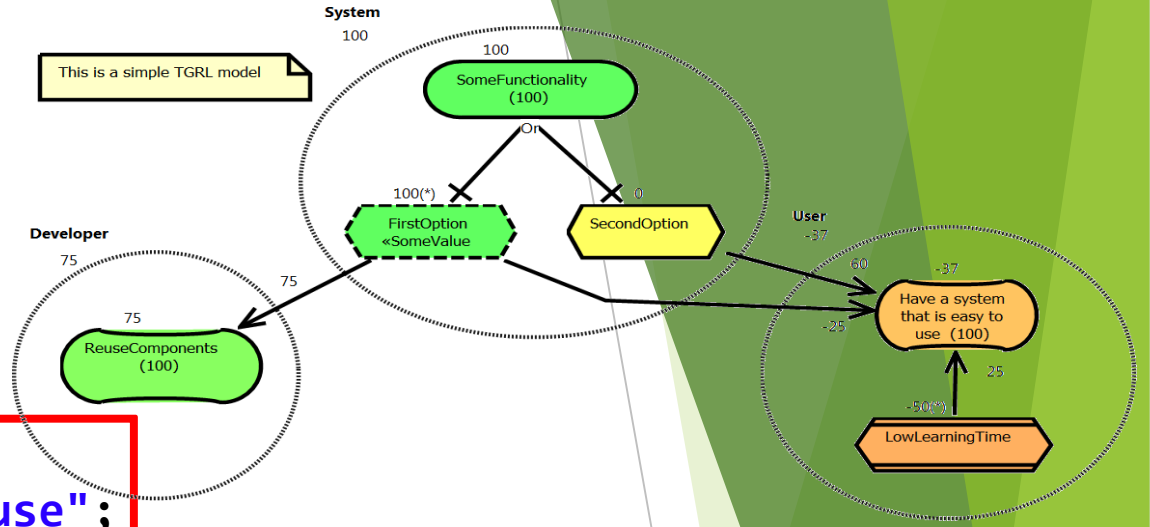


TGRl: Actors

```
grl IStar2015{

    actor User;

    actor Developer { }

    actor System {
    }

}
```

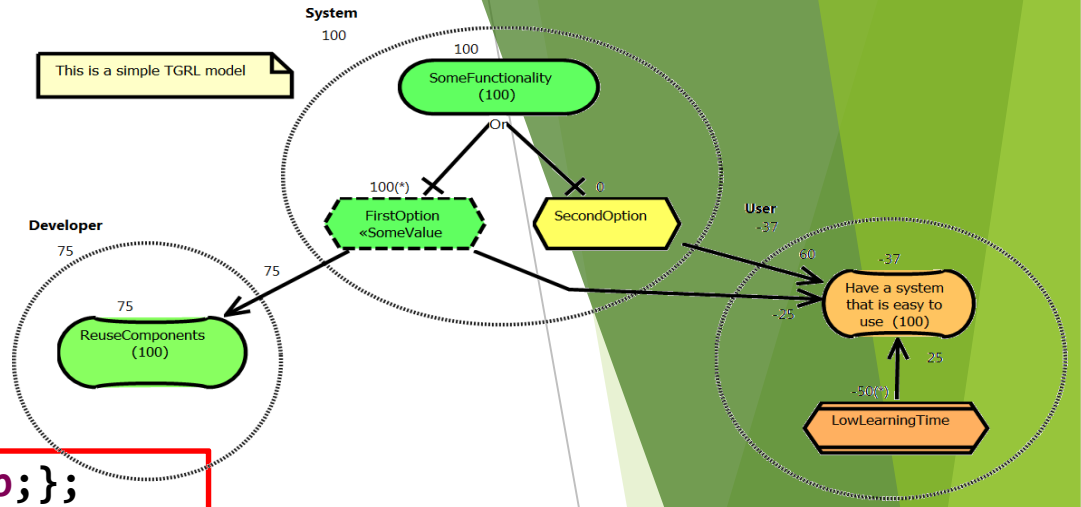


TGRl: Intentional Elements



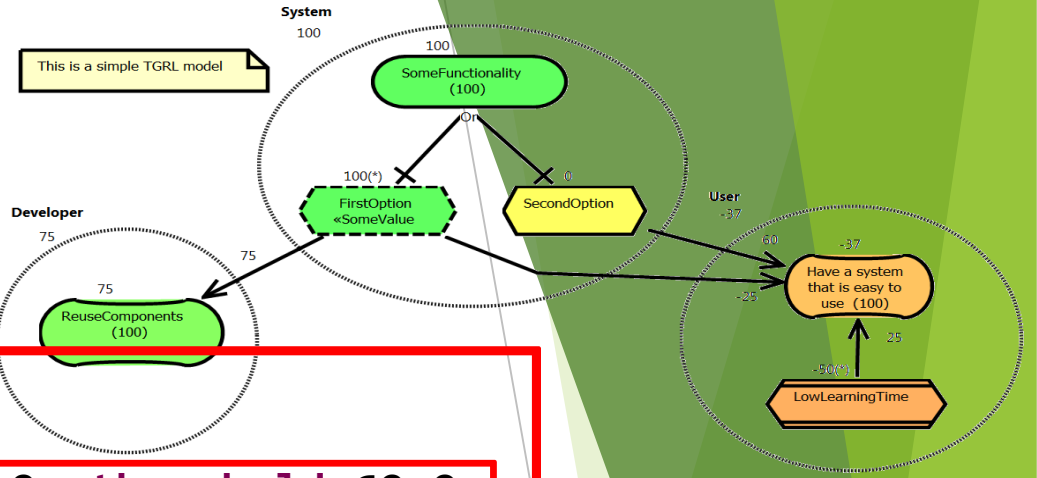

```
grl IStar2015 {
    actor User {
        softGoal EasyToUse {
            name="Have a system that is easy to use";
            importance = 100;
        }
        indicator LowLearningTime; // Indicator definition
    }
    actor Developer {
        softGoal ReuseComponents {importance=100;}
    }
    actor System {
        goal SomeFunctionality {importance=high; decompositionType=or;}
        task FirstOption {metadata stereotype="SomeValue";}
        task SecondOption {description = "Better alternative";}
    }
}
```

TGRl: Element Links




```
actor User {
    softGoal EasyToUse; indicator LowLearningTime;

    LowLearningTime contributesTo EasyToUse {name=C2;help;};
}
actor Developer {softGoal ReuseComponents;   }
actor System {
    goal SomeFunctionality;   task FirstOption;    task SecondOption;

    SomeFunctionality decomposedBy FirstOption, SecondOption;
    FirstOption contributesTo Developer.ReuseComponents {75;};
}

System.FirstOption contributesTo User.EasyToUse {hurt;};
System.SecondOption contributesTo User.EasyToUse {name=C1;60;};

link mustUse; // Link type definition
User mustUse System; // Link instance between two actors
```

TGRl: Evaluation Strategies

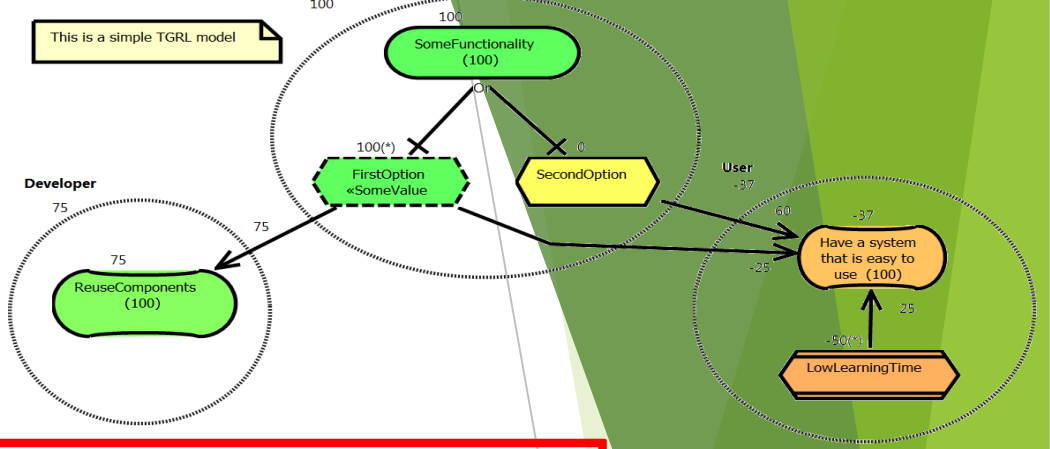



```
strategy SelectFirst {
    System.FirstOption = satisfied;
    User.LowLearningTime = {unit="minutes"; target=30.0; threshold=60.0;
        worst=120.0; eval=90.0;}
}
```

```
strategy SelectSecond extends SelectFirst { // Strategy inclusion!
    System.FirstOption = none; // Overridden
    System.SecondOption = 100; // Added, quantitatively this time
}

strategy RangeExample extends SelectFirst {
    System.FirstOption = {start = 10; end = 40; step = 5;}
}
```

```
strategyGroup MyGroup includes SelectFirst, SelectSecond, RangeExample;
```

TGRl: Contribution overrides

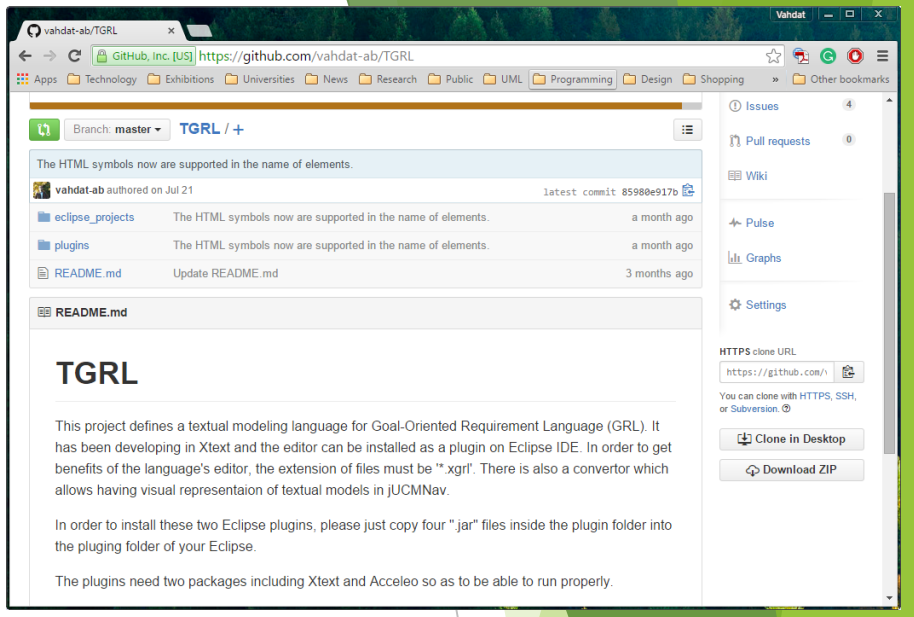



```
contribution FirstOverride {
    System.C1 = 30; User.C2 = make;
}
```

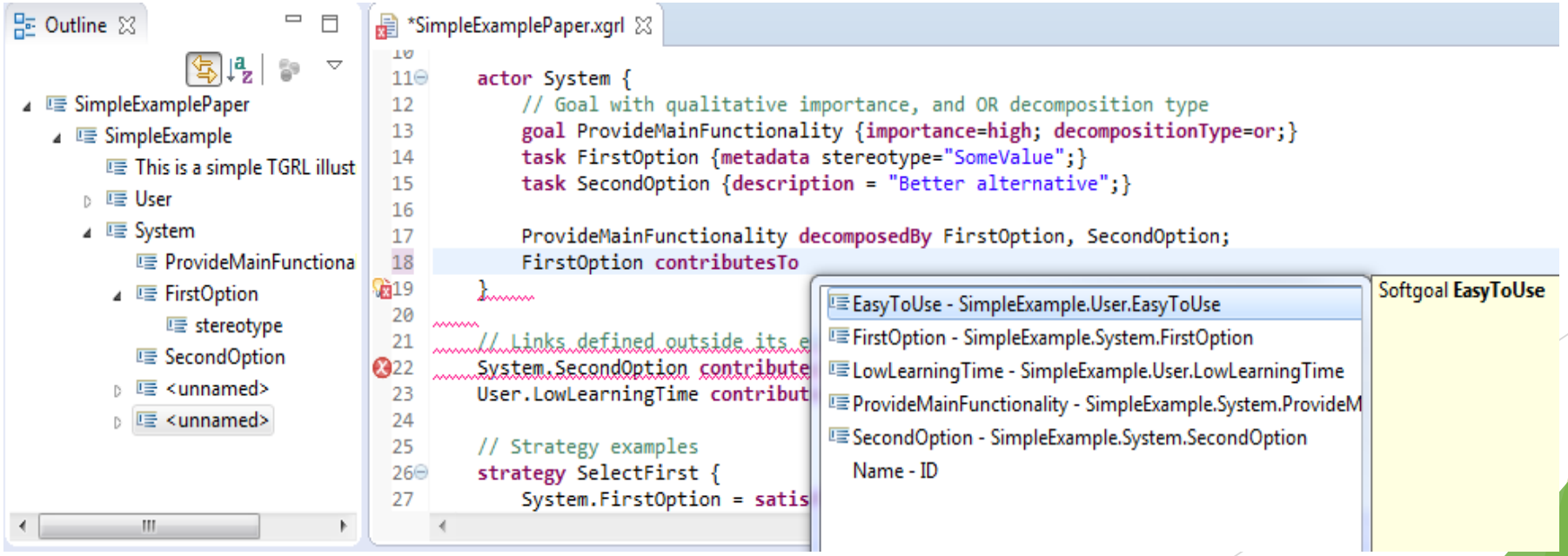
```
contribution SecondOverride extends FirstOverride {
      System.C1 = {start = -40; end = 0; step = 10;}
}
```

```
contributionGroup SomeOverrides includes FirstOverride,
SecondOverride;
```

TGRL Editor and Transformation



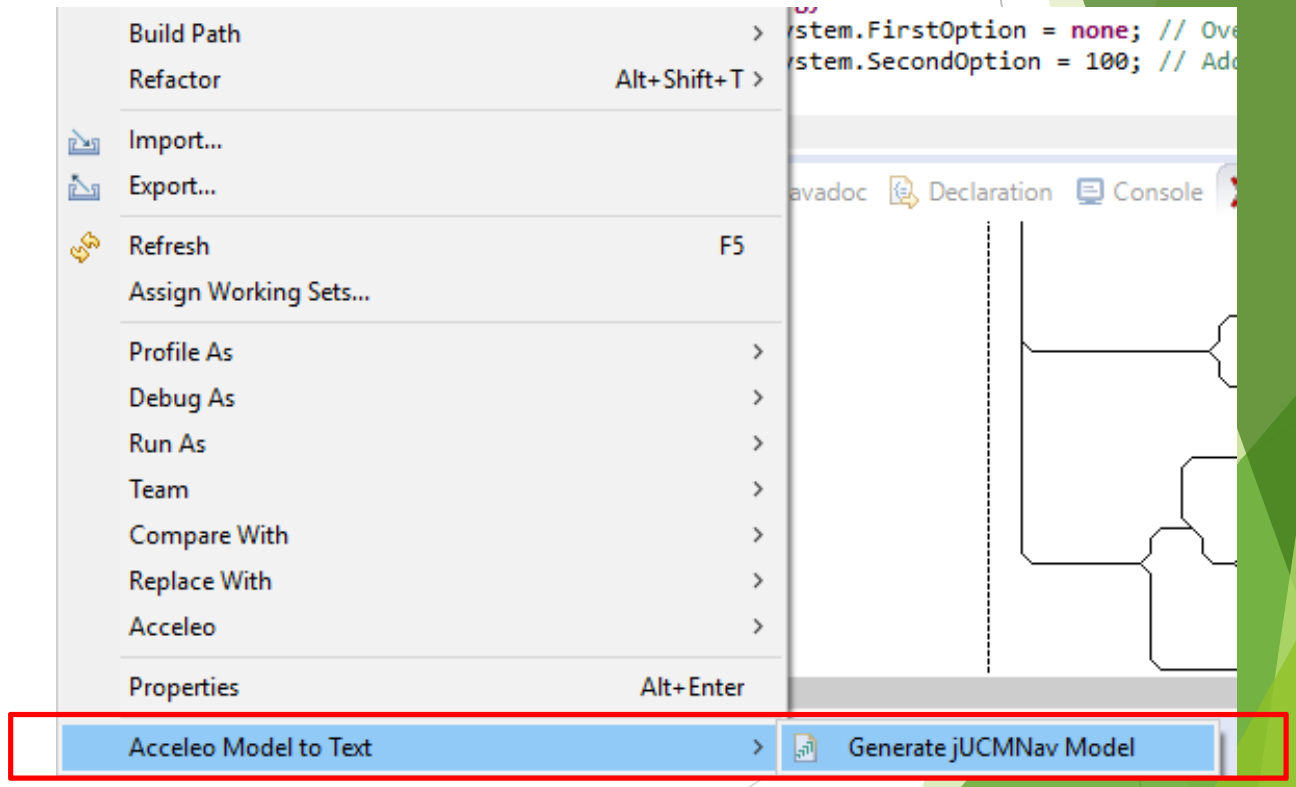
- Eclipse plugin
- Open source (https://github.com/vahdat-ab/TGRL)
- Developed with Xtext
- Supports syntax highlight, an outline view, annotation of syntactic errors, content assistance, and code formatting



uOttawa

TGRL Editor and Transformation cont.

- Transforms TGRL models to URN/jUCMNav models
- Model-to-code transformation (implemented with Acceleo)
- Not handling the layout



Conclusion

- Illustrated a new textual syntax for GRL, called TGRL, with a full coverage of the language.
- Covered many concepts such as indicators, strategies, contribution overrides, metadata and URN links.
- Developed a feature-rich Eclipse-based editor.
- Implemented an automated conversion to GRL models readable by jUCMNav.

Future Work

- The language and the tool require further and more rigorous validation.
- Improving by the inclusion of additional static semantic rules to ensure the correctness of the GRL models
- A transformation from jUCMNav to TGRL (or develop a synchronized textual and graphical tool)
- Combine TGRL (for goals) with Umple (for design and implementation) as they provide complementary concepts
- Extend this language to support the whole URN standard

Thank You for your Attention