

# GrowingLeaf: Supporting Requirements Evolution over Time

Alicia M. Grubb, Gary Song, Marsha Chechik  
{amgrubb, gary, chechik}@cs.toronto.edu

iStar'16 - Sep. 12, 2016



# Problem

---

- Assumptions of early-phase requirements modeling:
  - ➔ all model elements have a value
  - ➔ model values are constant
- In reality intentions and relationships in the environments are not constant.

# Example Questions

---

1. Is it possible to satisfy *Goal-A* and partially satisfy *Goal-B*? and how?
2. How does completing *Task-A* and *Task-B* but not *Task-C* affect the top level goals?
3. How do changes in *Actor-A*'s dependums affect the *Actor-A*'s root-level goals over time?
4. Which possible scenarios always satisfy *Goal-A* even if *Goal-B* becomes denied in the future?
5. Does the satisfaction order of *Goal-C* and *Goal-D* matter?

# Example Questions

---

1. Is it possible to satisfy *Goal-A* and partially satisfy *Goal-B*? and how?
2. How does completing *Task-A* and *Task-B* but not *Task-C* affect the top level goals?
3. How do changes in *Actor-A*'s dependums affect the *Actor-A*'s root-level goals over time?
4. Which possible scenarios always satisfy *Goal-A* even if *Goal-B* becomes denied in the future?
5. Does the satisfaction order of *Goal-C* and *Goal-D* matter?

# Example Questions

---

1. Is it possible to satisfy *Goal-A* and partially satisfy *Goal-B*? and how?
2. How does completing *Task-A* and *Task-B* but not *Task-C* affect the top level goals?
3. How do changes in Actor A affect the Actor A?
4. Which method with Forward Analysis and Backward Analysis is better for Goal-A even if Goal-A is carried in the future?
5. Does the satisfaction order of *Goal-C* and *Goal-D* matter?

Use Qualitative Evaluation Labels with Forward Analysis and Backward Analysis

# Example Questions

---

1. Is it possible to satisfy *Goal-A* and partially satisfy *Goal-B*? and how?
2. How does completing *Task-A* and *Task-B* but not *Task-C* affect the top level goals?
3. How do changes in *Actor-A*'s dependums affect the *Actor-A*'s root-level goals over time?
4. Which possible scenarios always satisfy *Goal-A* even if *Goal-B* becomes denied in the future?
5. Does the satisfaction order of *Goal-C* and *Goal-D* matter?

# Example Questions

---

1. Is it possible to satisfy *Goal-A* and partially satisfy *Goal-B*? and how?
2. How does completing *Task-A* and *Task-B* but not *Task-C* affect the top level goals?
3. How do changes in *Actor-A*'s dependums affect the *Actor-A*'s root-level goals over time?
4. Which possible scenarios always satisfy *Goal-A* even if *Goal-B* becomes denied in the future?
5. Does the satisfaction order of *Goal-C* and *Goal-D* matter?

# Example Questions

---

1. Is it possible to satisfy *Goal-A* and partially satisfy *Goal-B*? and how?
2. How does completing *Task-A* and *Task-B* but not *Task-C* affect the top level goals?
3. How do changes in *Actor-A*'s dependums affect the *Actor-A*'s root-level goals over time?
4. Which possible scenarios always satisfy *Goal-A* even if *Goal-B* becomes denied in the future?
5. Does the satisfaction order of *Goal-C* and *Goal-D* matter?

# Example Questions

---

1. Is it possible to satisfy *Goal-A* and partially satisfy *Goal-B*? and how?
2. How does completing *Task-A* and *Task-B* but not *Task-C* affect the top level goals?
3. How do changes in *Actor-A*'s dependums affect the *Actor-A*'s root-level goals over time?
4. Which possible scenarios always satisfy *Goal-A* even if *Goal-B* becomes denied in the future? 
5. Does the satisfaction order of *Goal-C* and *Goal-D* matter?

# Contributions

---

Provide tooling to:

- enrich goal models intentions with dynamically changing evaluation
- analyze the impacts of dynamically changing intentions on decision making

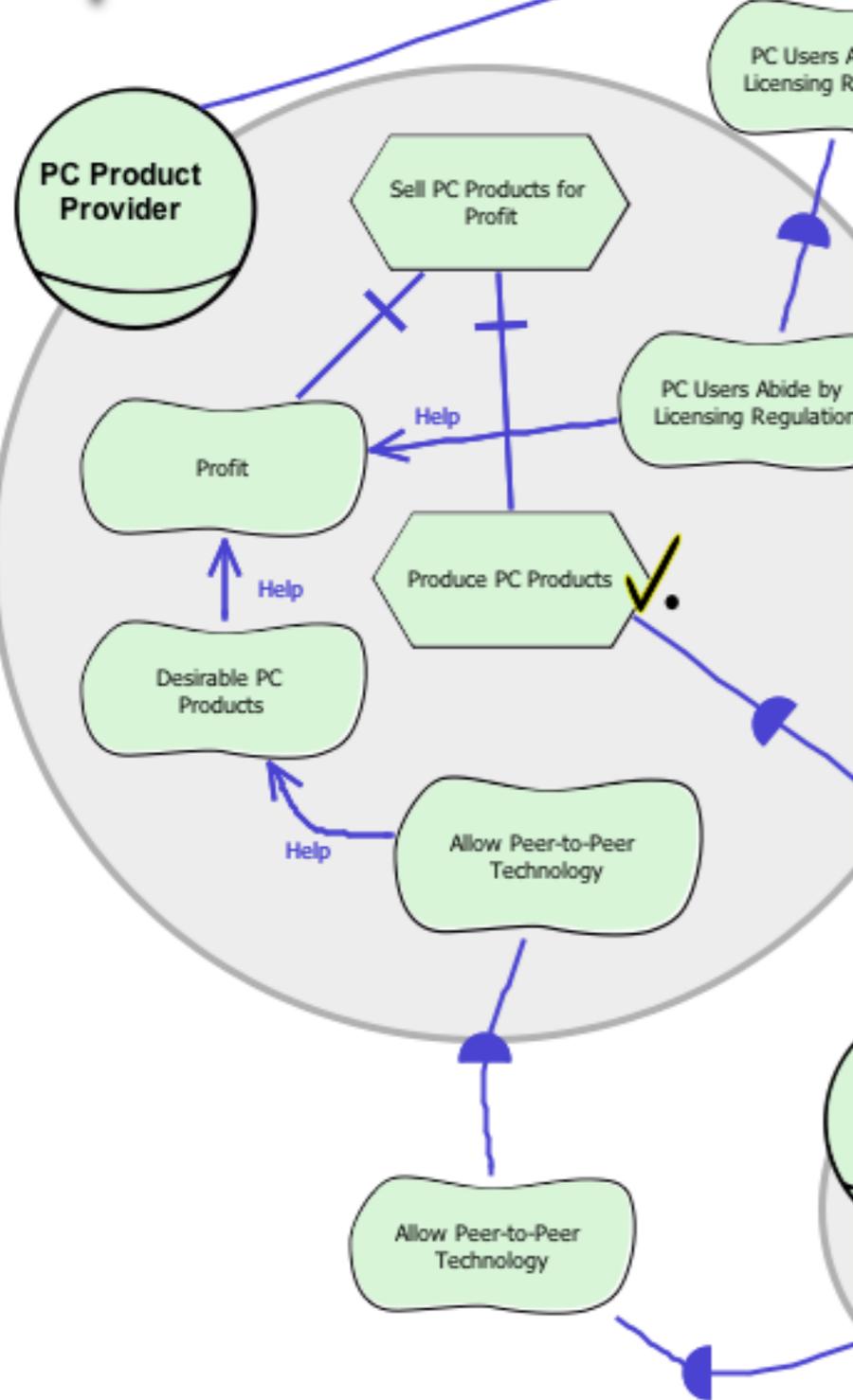
# Why another modeling tool?

- Why another modeling tool?



# Why another modeling tool?

## OpenOME



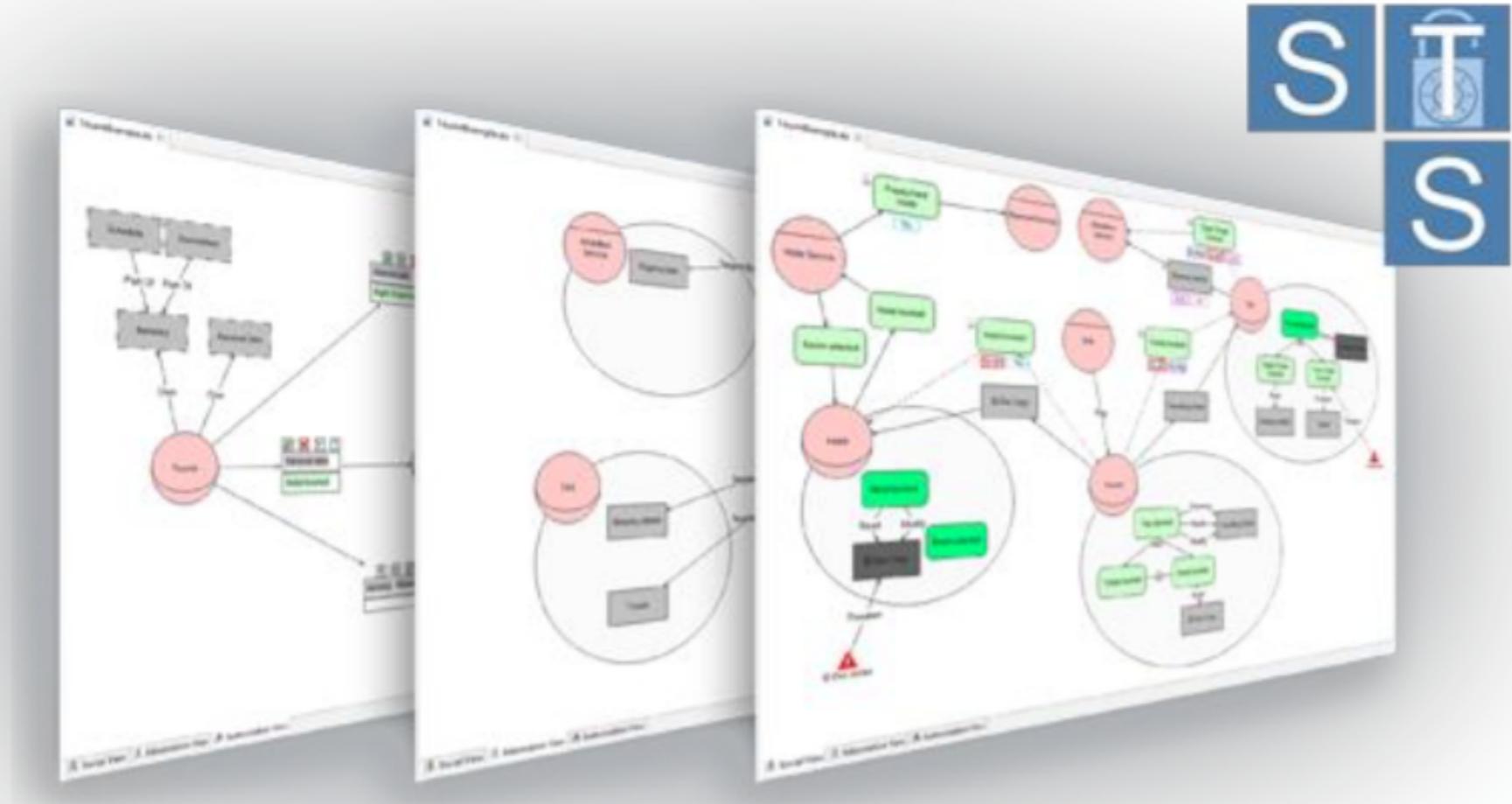
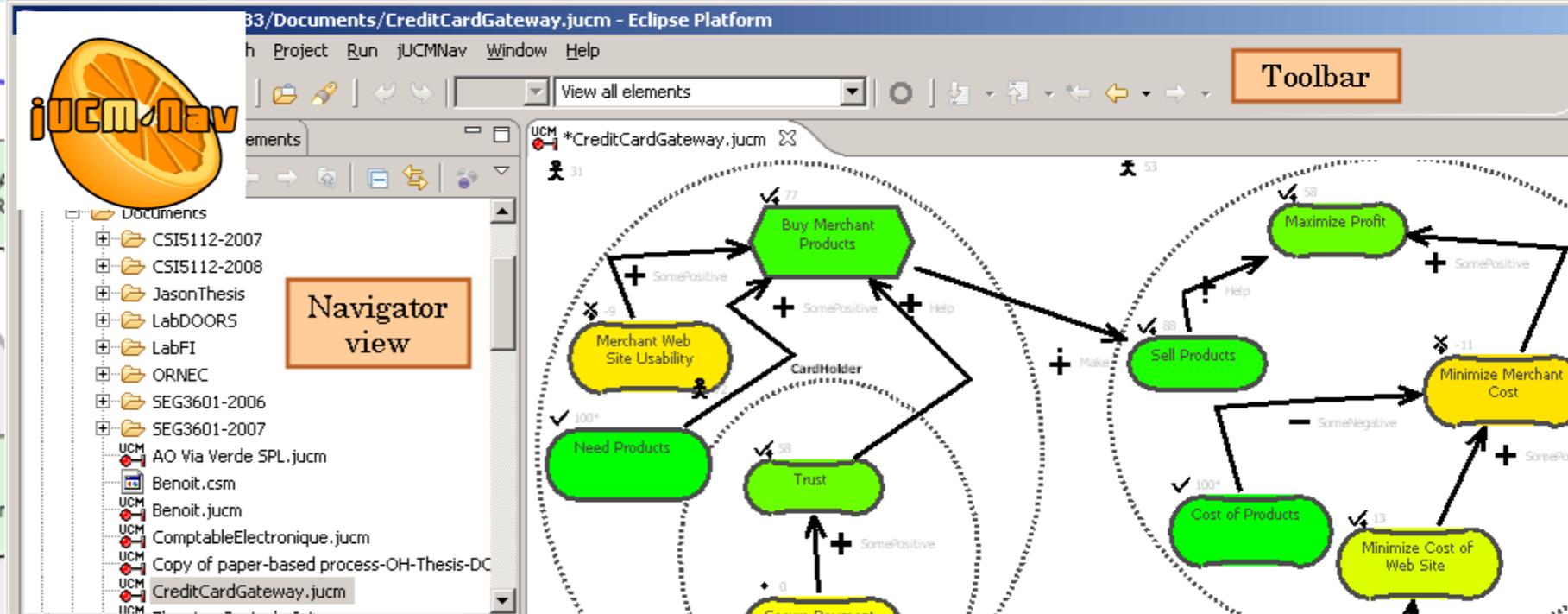
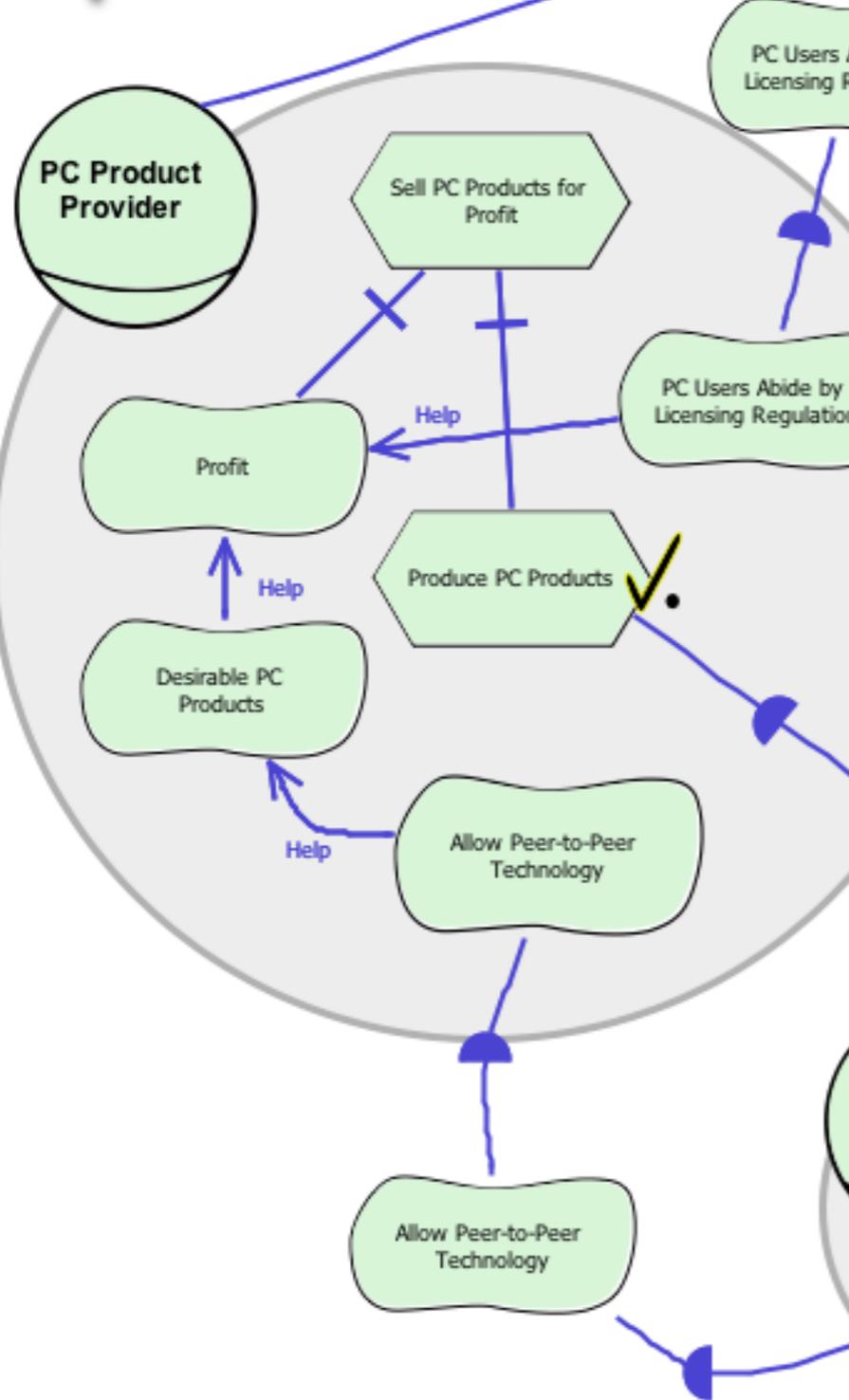
The screenshot displays the iUCMNav Eclipse Platform interface for the project **CreditCardGateway.jucm**. The interface includes several key views:

- Navigator view:** Shows a tree of documents and UCM elements.
- Outline view:** Lists UCM elements with their IDs, such as **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)**, and **Cost (6259)**.
- Editor:** Displays a Goal Structuring Notation (GSN) diagram with nodes like **Buy Merchant Products**, **Merchant Web Site Usability**, **Need Products**, **Trust**, **Secure Payment**, **Sell Products**, **Maximize Profit**, **Minimize Merchant Cost**, **Cost of Products**, **Minimize Cost of Web Site**, and **Minimize Implementation Cost**. Relationships are labeled with **SomePositive**, **Help**, **Make**, **SomeNegative**, and **+**.
- Scenario and Strategies view:** Shows a tree of scenarios and strategies, including **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**, and **Third Party Payment**.
- Property view:** Shows a table of properties for the selected element.

Property	Value
<b>Info</b>	
description	
id	3030
name	Minimize Implementation
Parent	Payment Gateway (3012)
<b>Metadata</b>	
Metadata	[click to edit]
<b>Miscellaneous</b>	
criticality	None
decompositionType	And

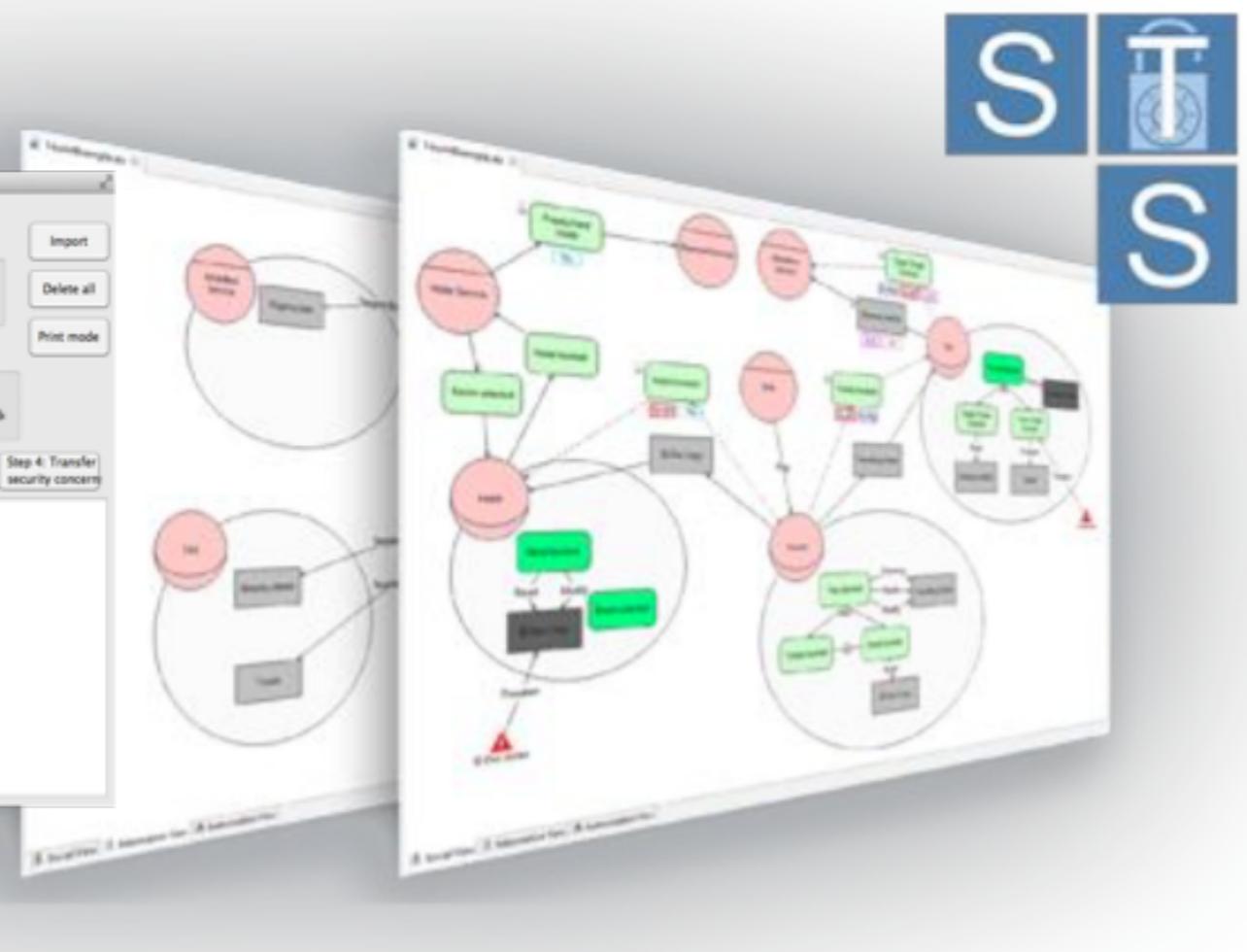
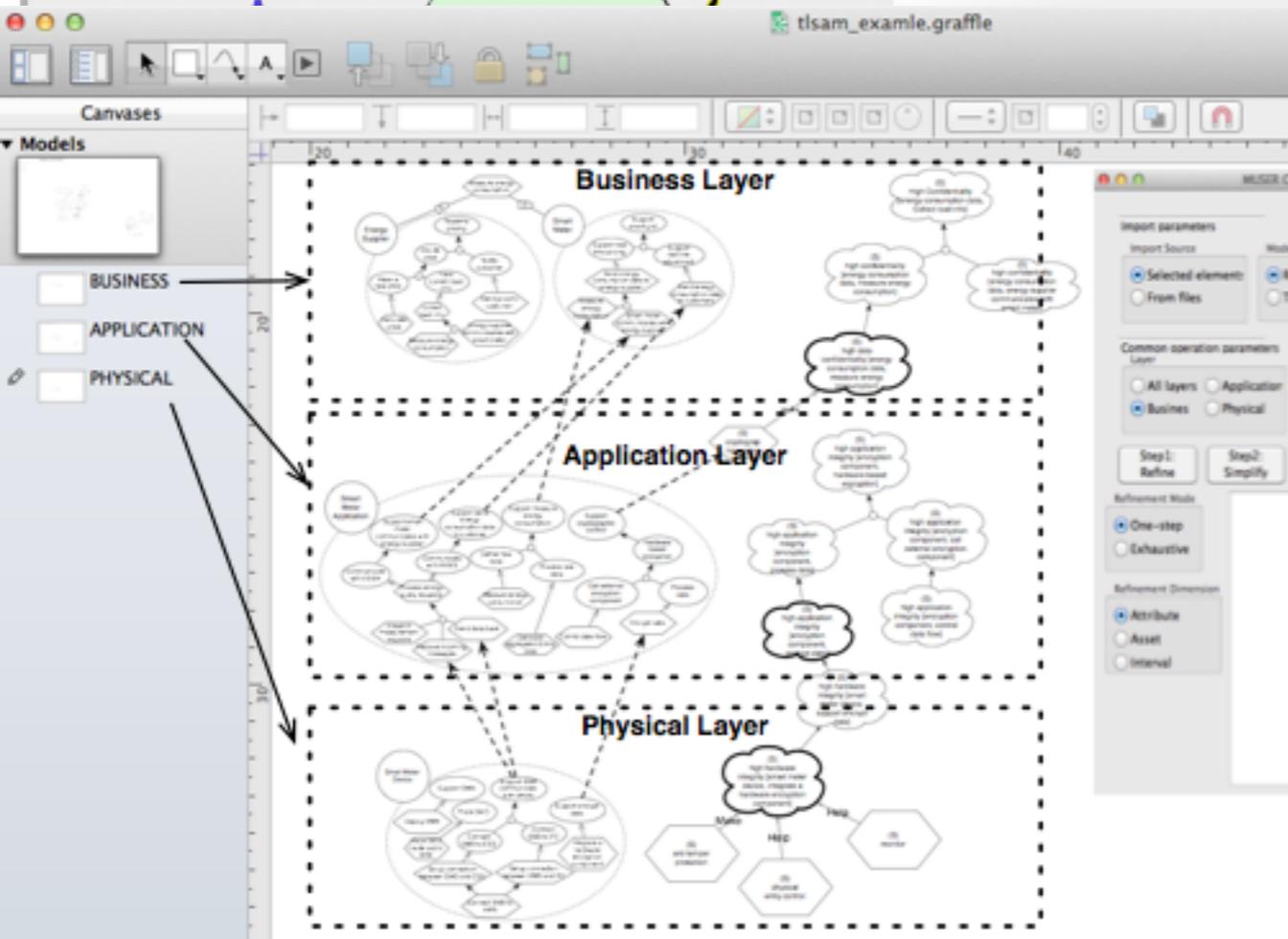
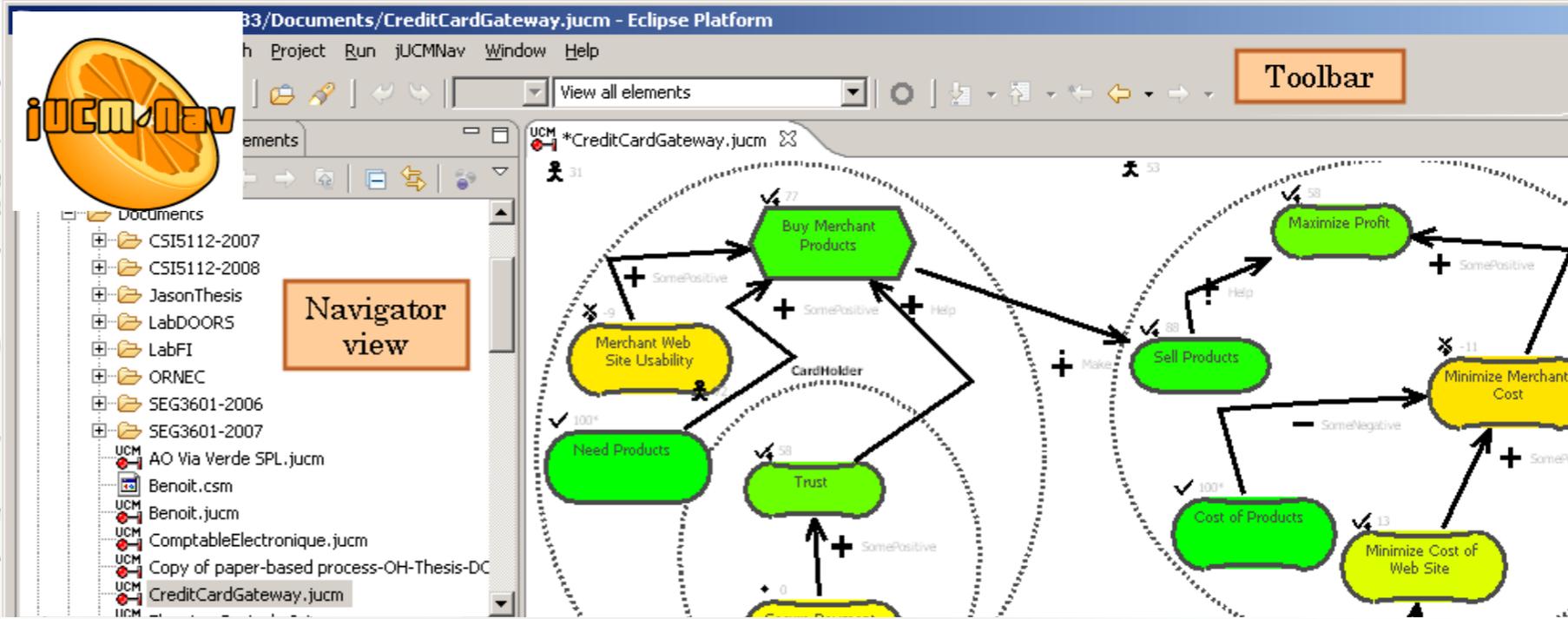
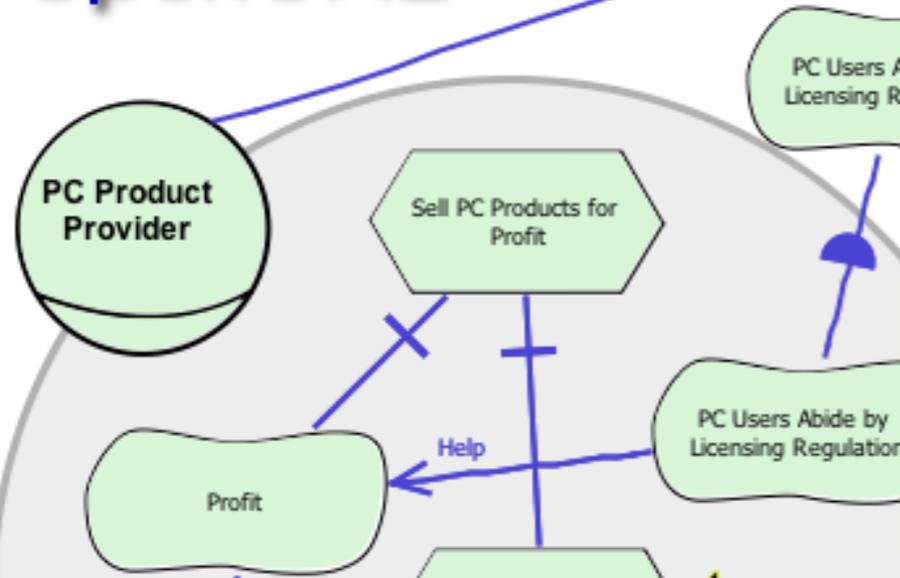
# Why another modeling tool?

## OpenOME



# Why another modeling tool?

## OpenOME



# Why another modeling tool?

- Surveyed previous tools
  - Extend their iStar meta-model
  - Add icons/labels on top of intentions
- Web-based tool
  - Framework vs. self-built
  - Multi-view vs. multi-tab

# Introducing GrowingLeaf

www.cs.utoronto.ca/~amgrubb/leaf-ui/Tool.html

**GrowingLeaf** Undo Redo Clear Save Load Zoom In Zoom Out Open as SVG Export .leaf Font Size Model Constraints Analysis

Stencil

- Goal (Yellow rounded rectangle)
- Task (Green hexagon)
- Soft Goal (Orange rounded rectangle)
- Resource (Blue rounded rectangle)
- Actor (Yellow circle)

Modelling Relationships

Node name: Build Green Centre

Initial Satisfaction Value: Denied

Function Type: Denied Satisfied

Copyright 2015-2016. University of Toronto Department of Computer Science. All rights reserved.

Powered by: client IO. All rights reserved. JointJS: an HTML 5 diagramming component. <http://jointjs.com>

# Introducing GrowingLeaf

The screenshot displays the GrowingLeaf web application interface. At the top, the browser address bar shows the URL [www.cs.utoronto.ca/~amgrubb/leaf-ui/Tool.html](http://www.cs.utoronto.ca/~amgrubb/leaf-ui/Tool.html). The application header includes the 'GrowingLeaf' logo and a menu with options: Undo, Redo, Clear, Save, Load, Zoom In, Zoom Out, Open as SVG, Export .leaf, Font Size, Model Relationships, and Analysis. The 'Analysis' tab is currently selected.

On the left side, there is a 'Stencil' panel with various modeling elements: Goal (yellow rounded rectangle), Task (green hexagon), Soft Goal (orange rounded rectangle), Resource (blue rectangle), and Actor (yellow circle). Below the stencil is a copyright notice: 'Copyright 2015-2016. University of Toronto Department of Computer Science. All rights reserved.'

The main workspace is titled 'Modelling Constraints' and contains a diagram. The diagram features a central actor 'City' (yellow circle) connected to several tasks (green hexagons) and soft goals (orange rounded rectangles). The tasks include 'Update Truck Route DS', 'Upgrade Trucks DS', 'Use Current Dump', 'Use New Dump', 'Build Large Dump DS', 'Build Small Dump C', 'Purchase Land DS', 'Process Green Waste', 'Build Green Centre DS', 'GW Education Program UD', and 'Have Workers Union Contract UD'. The soft goals include 'Manage City Waste', 'Reduce Operating Costs', 'Comply with Standards C', and 'Positive City Image'. The 'Space in Dump MN' resource is also present. Red 'X' marks indicate failed or invalid elements, while green checkmarks indicate successful or valid elements. Small '<' symbols are placed near some task-to-task relationships.

On the right side, there is an 'Intension Relationship' panel with two dropdown menus, both currently showing 'A'.

At the bottom left, there is a 'Powered by:' section with logos for 'client IO' and 'JointJS: an HTML 5 diagramming component', along with the URL <http://jointjs.com>.

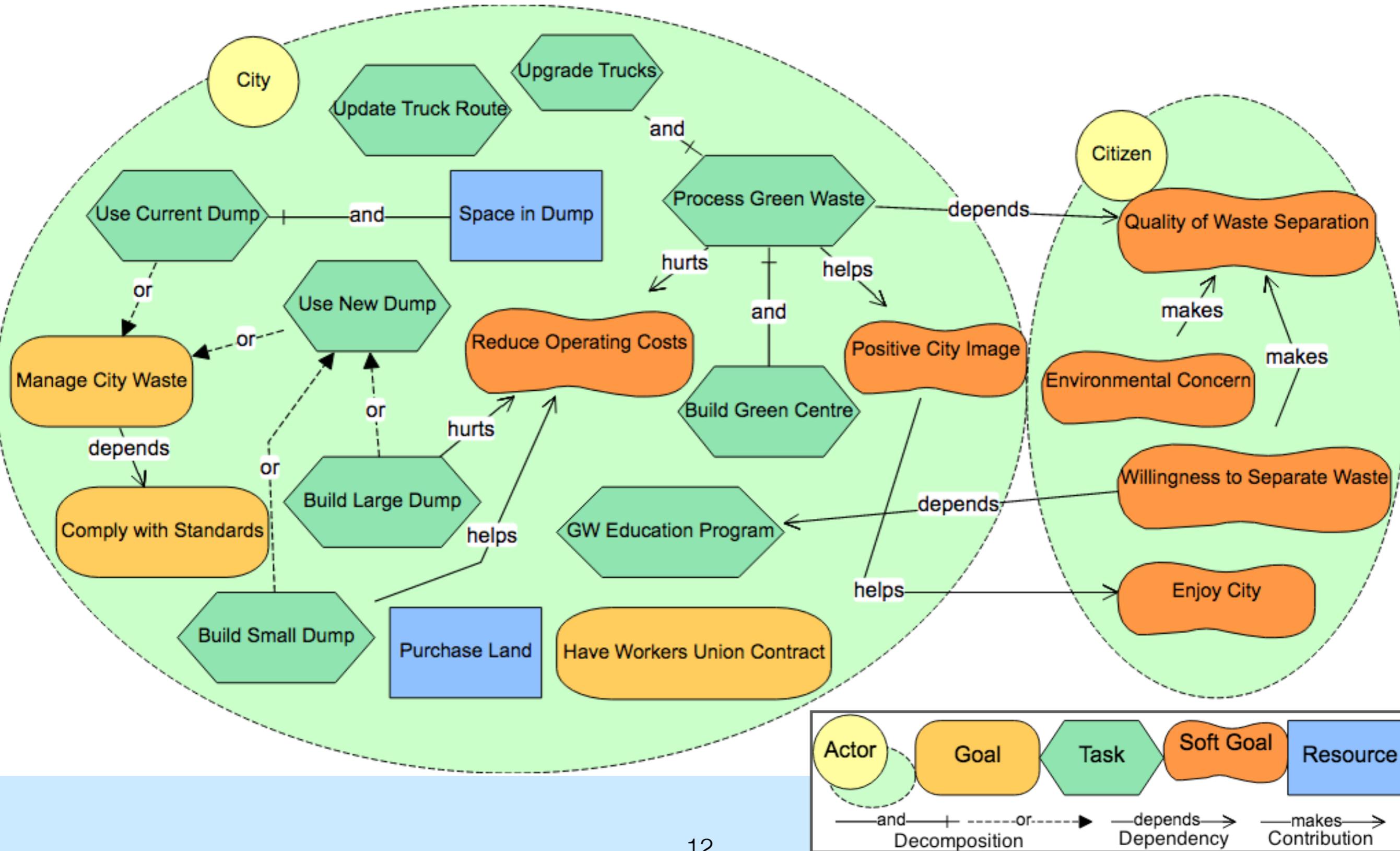


# Outline

---

- Modeling Problem and Tool Justification
- Tool Introduction
- **Dynamic Intentions and Analysis**
- Tool Functionality
- Discussion and Validation
- Status and Future Work

# Modeling Dynamic Intentions





# Modeling Dynamic Intentions

---

## Stochastic (R)

Patterns:



Examples:



# Modeling Dynamic Intentions

---

## Elementary Functions

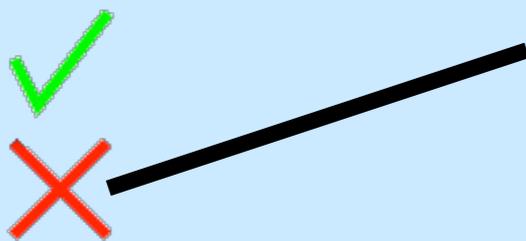
Stochastic (R):



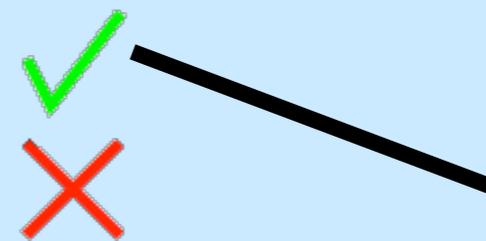
Constant (C):



Increase (I):



Decrease (D):

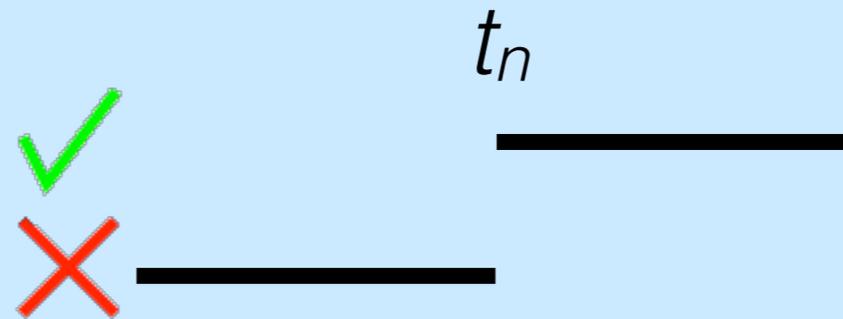


# Modeling Dynamic Intentions

---

## Denied-Satisfied (DS)

Patterns:



Examples:



# Modeling Dynamic Intentions

## Denied-Satisfied (DS)

Patterns:



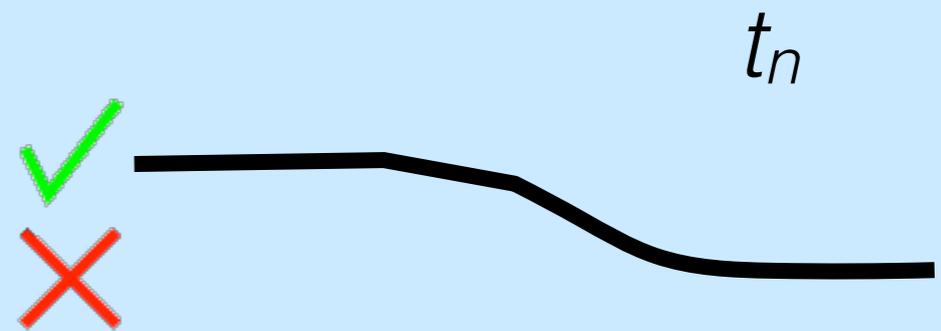
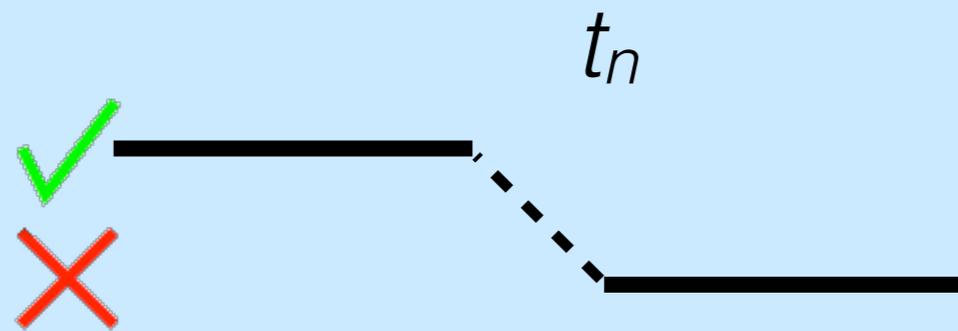
Examples:



# Modeling Dynamic Intentions

## Monotonic Negative (MN)

Patterns:



Examples:



# Common Compound Functions

---

Denied-Satisfied  
(DS)

the satisfaction evaluation remains *Denied* until  $t_i$  and then remains *Satisfied*

Monotonic Negative  
(MN)

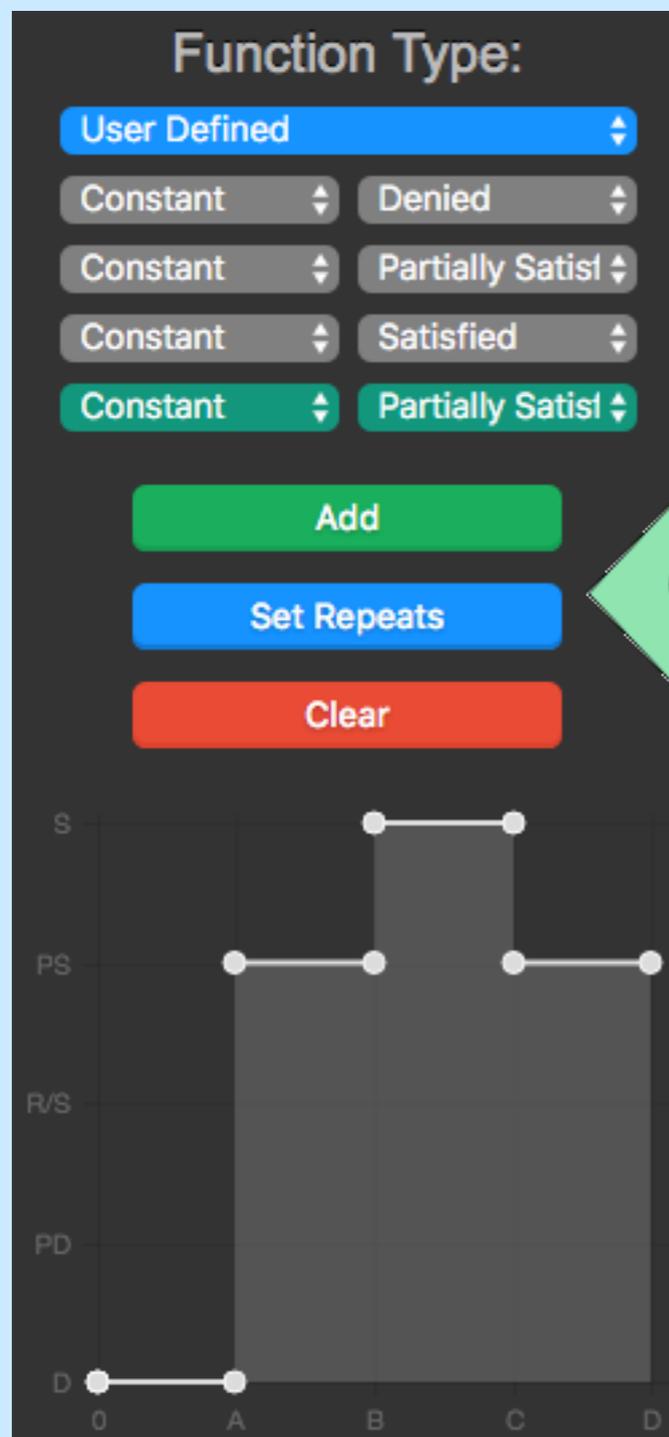
changes in satisfaction evaluation become “less true” to a *maxValue* at  $t_i$  and then remains constant at *constantValue*

# Common Compound Functions

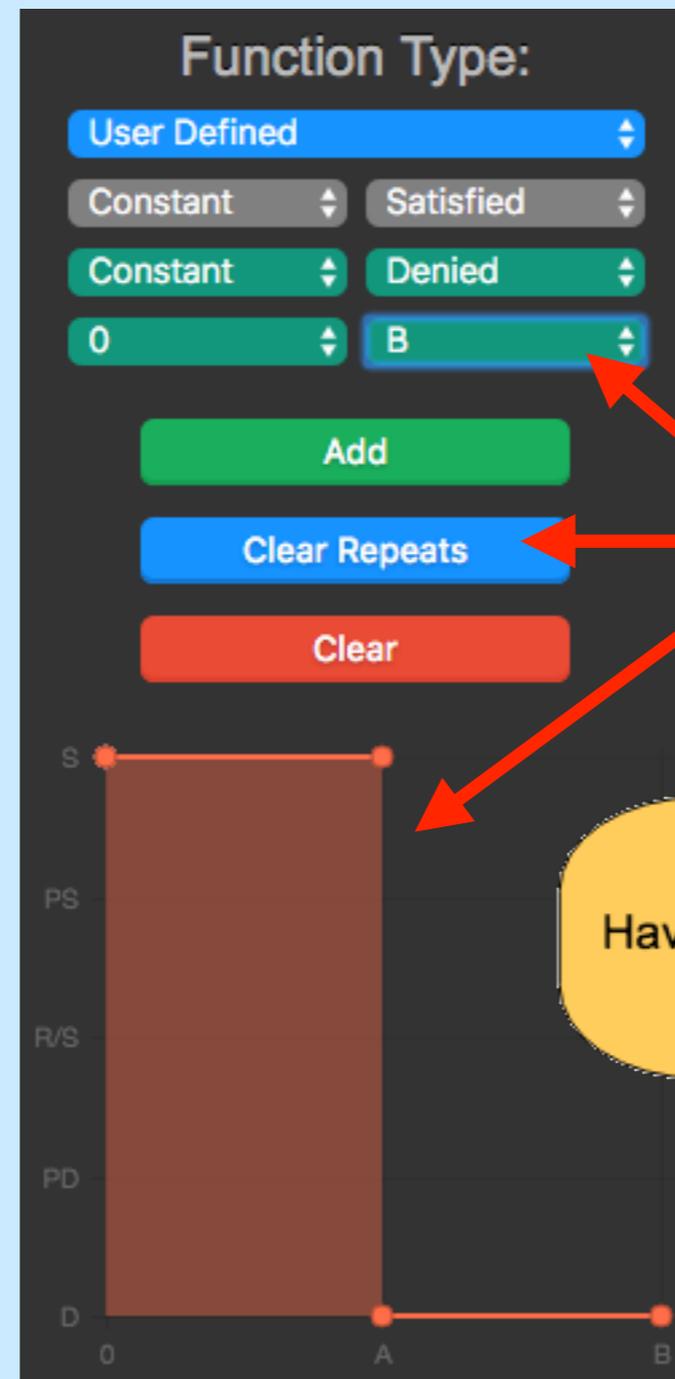
Satisfied-Denied (SD)	the satisfaction evaluation remains <i>Satisfied</i> until $t_i$ and then remains <i>Denied</i>
Denied-Satisfied (DS)	the satisfaction evaluation remains <i>Denied</i> until $t_i$ and then remains <i>Satisfied</i>
Stochastic-Constant (RC)	changes in satisfaction evaluation are stochastic or random until $t_i$ and then remains constant at <i>constantValue</i>
Constant-Stochastic (CR)	the satisfaction evaluation remains constant at <i>constantValue</i> until $t_i$ and then changes in evaluation are stochastic or random
Monotonic Positive (MP)	changes in satisfaction evaluation become “more true” to a <i>maxValue</i> at $t_i$ and then remains constant at <i>constantValue</i>
Monotonic Negative (MN)	changes in satisfaction evaluation become “less true” to a <i>maxValue</i> at $t_i$ and then remains constant at <i>constantValue</i>

# Modeling Dynamic Intentions

## User Defined (UD)



GW Education Program



Repeating Function

Have Workers Union Contract

# Analysis Strategies

---

(Strategy 1: Leaf Simulation) create a **random path** given initial states in the model

(Strategy 2: CSP Analysis) create a path given **desired properties** of the **intermediate state** (with optional properties over the initial or final state)

(Strategy 3: CSP History) create a path which is **different than the previously seen path** over the same constraints

# Outline

---

- Modeling Problem and Tool Justification
- Tool Introduction
- Dynamic Intentions and Analysis
- **Tool Functionality**
- Discussion and Validation
- Status and Future Work

# GrowingLeaf - Modeling Demo

**GrowingLeaf**

Undo Redo

Clear

Save

Load

Zoom In

Zoom Out

Open as SVG

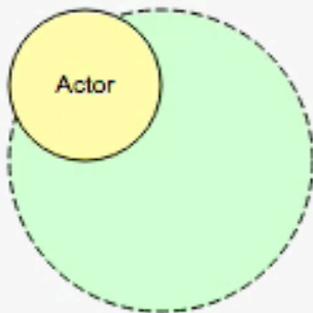
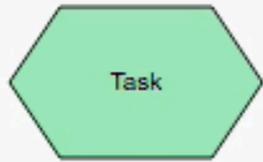
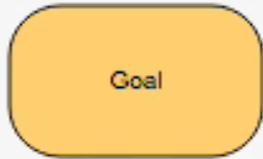
Export .leaf

Font Size

Model Constraints

Analysis

Stencil



Modelling Relationships



Copyright 2015-2016.  
University of Toronto  
Department of Computer Science.  
All rights reserved.

Powered by:



Copyright 2014-2016.  
client IO. All rights reserved.  
JointJS: an HTML 5 diagramming  
component.  
<http://jointjs.com>

# GrowingLeaf - Modeling Demo

**GrowingLeaf**

Undo Redo

Clear

Save

Load

Zoom In

Zoom Out

Open as SVG

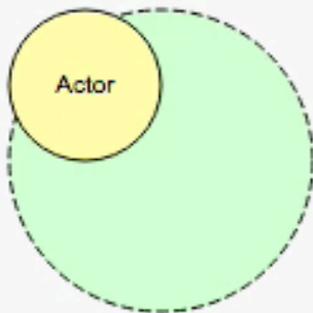
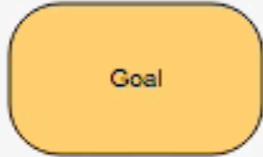
Export .leaf

Font Size

Model Constraints

Analysis

Stencil



Modelling Relationships

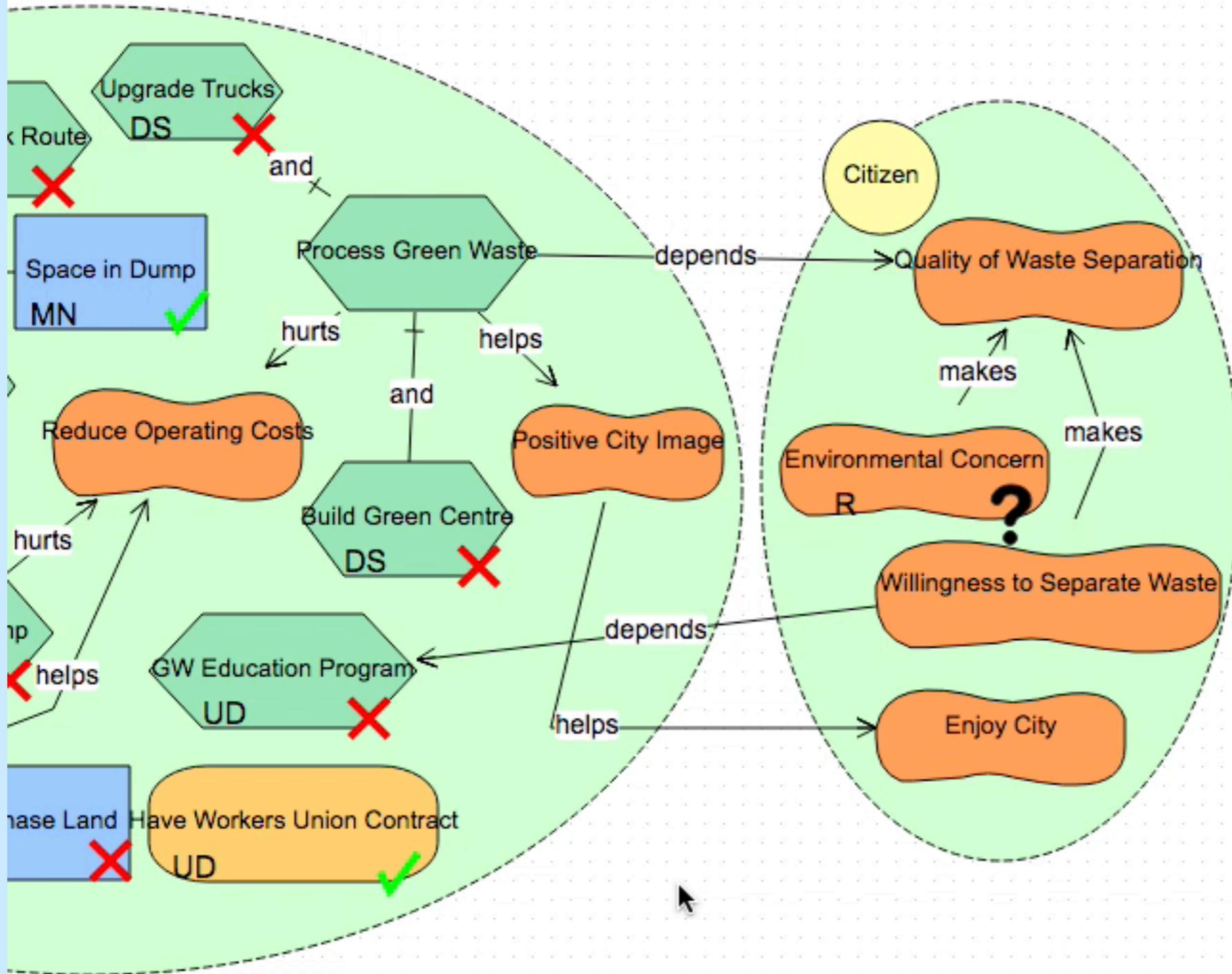


Copyright 2015-2016.  
University of Toronto  
Department of Computer Science.  
All rights reserved.

Powered by:



Copyright 2014-2016.  
client IO. All rights reserved.  
JointJS: an HTML 5 diagramming  
component.  
<http://jointjs.com>



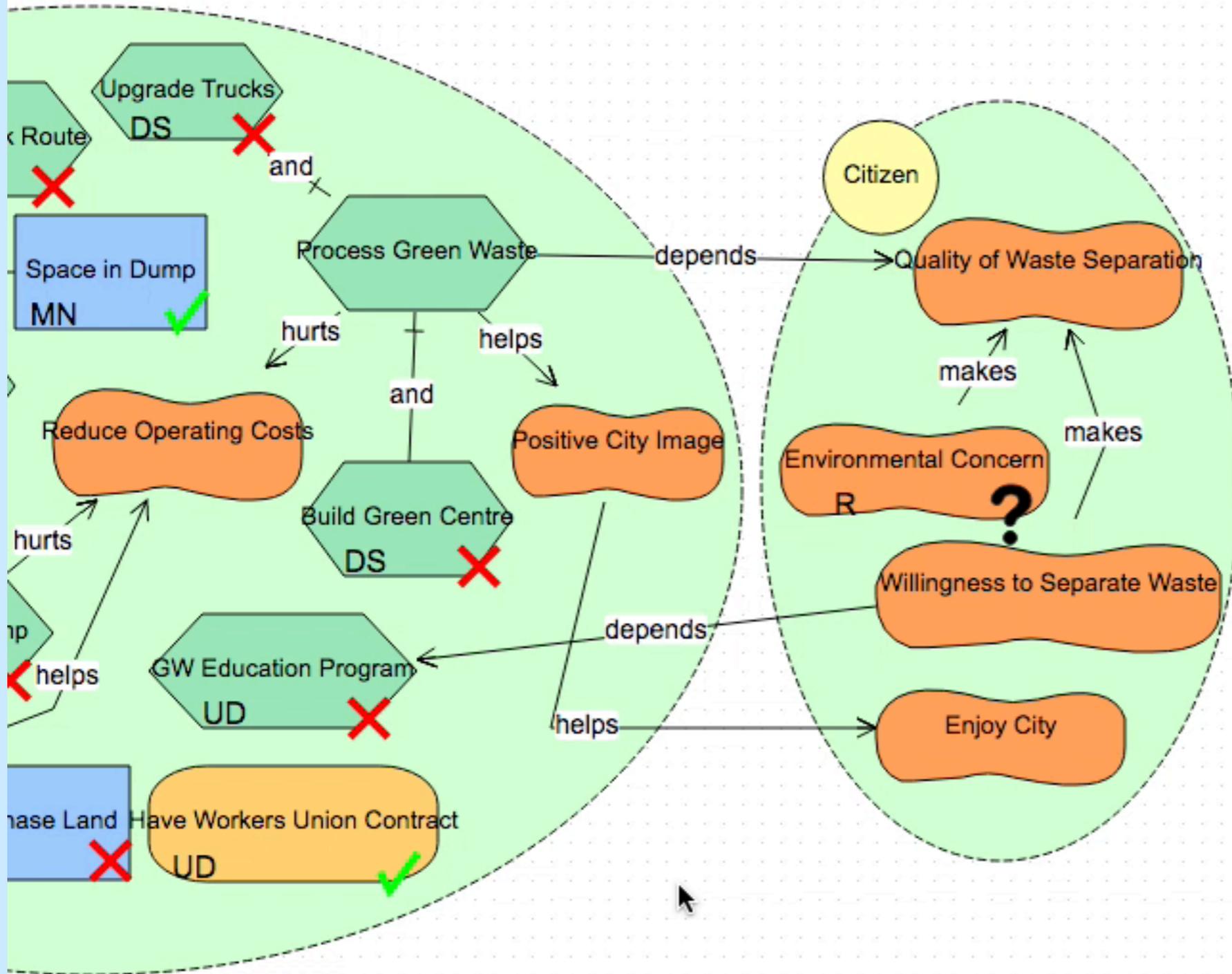
Node name:

Positive City Image

Initial Satisfaction Value:

None

# Modelling Relationships



Node name:

Positive City Image

Initial Satisfaction Value:

None

# GrowingLeaf - Modeling Demo Summary

---

- Drag and drop interface
- Naming and adding elements
- Loading, saving, exporting, and zooming models
- Resizing label fonts
- Changing initial satisfaction values
- Changing dynamic function types
- Creating User Defined functions

# GrowingLeaf - Analysis Demo

**GrowingLeaf** Undo Redo Clear Save Load Zoom In Zoom Out Open as SVG Export .leaf Font Size Model Constraints Analysis

Stencil

- Goal (Orange rounded rectangle)
- Task (Green hexagon)
- Soft Goal (Orange rounded rectangle)
- Resource (Blue rectangle)
- Actor (Yellow circle)

Modelling Relationships

Copyright 2015-2016. University of Toronto Department of Computer Science. All rights reserved.

Powered by: client IO. All rights reserved. JointJS: an HTML 5 diagramming component. <http://jointjs.com>

# GrowingLeaf - Analysis Demo

**GrowingLeaf** Undo Redo Clear Save Load Zoom In Zoom Out Open as SVG Export .leaf Font Size Model Constraints Analysis

Stencil

- Goal (Orange rounded rectangle)
- Task (Green hexagon)
- Soft Goal (Orange rounded rectangle)
- Resource (Blue rectangle)
- Actor (Yellow circle)

Modelling Relationships

Copyright 2015-2016. University of Toronto Department of Computer Science. All rights reserved.

Powered by: client IO. All rights reserved. JointJS: an HTML 5 diagramming component. <http://jointjs.com>

# GrowingLeaf - Analysis Demo Summary

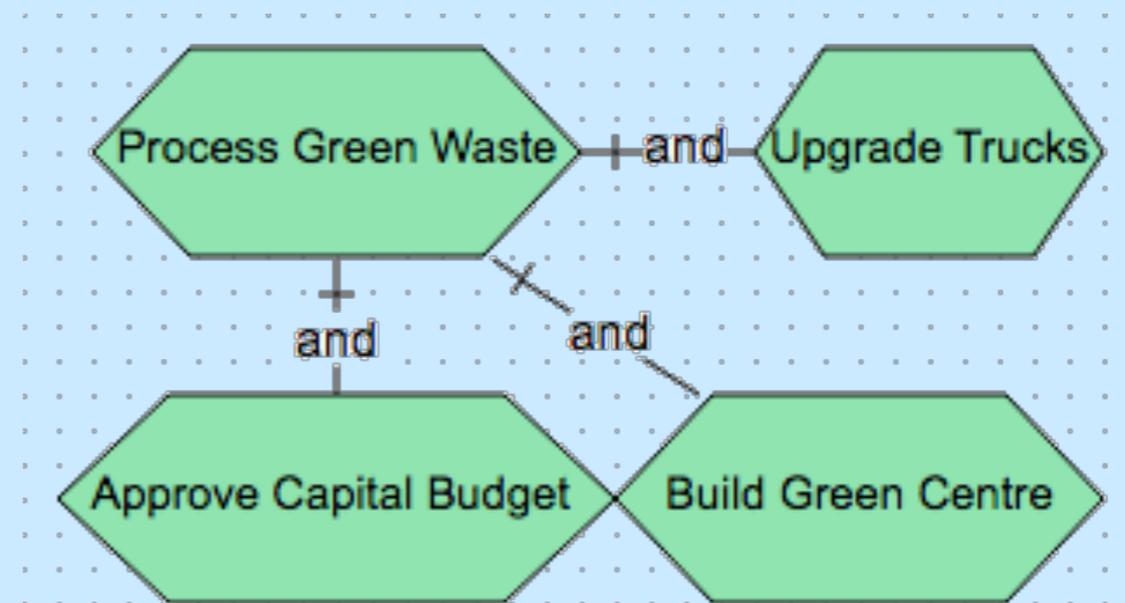
---

- How to run analysis
- Adjust simulation length
- Types of analysis
- Scrolling through analysis results

# Improving Analysis with Constraints

---

- Undesirable results due to EB ordering
- Add constraints over EB order
  - Adding model links is inappropriate
  - Test relationship before updating the model
- Used on rare occasions



# GrowingLeaf - Model Constraints Demo

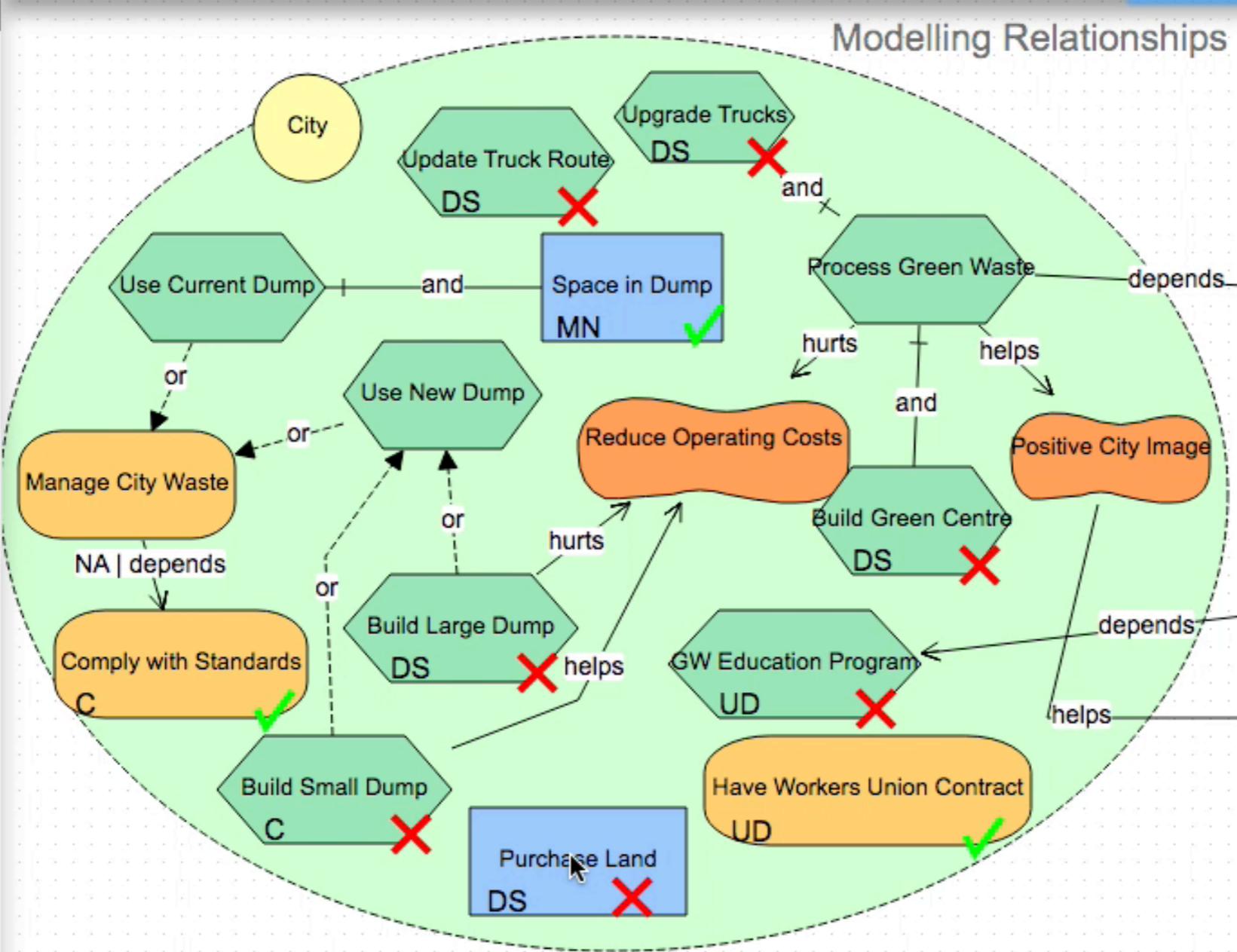
GrowingLeaf

Undo Redo Clear Save Load Zoom In Zoom Out Open as SVG Export .leaf Font Size

Model Constraints

Analysis

Stencil



Node name:

Comply with Standards

Initial Satisfaction Value:

Satisfied

Function Type:

Constant



# GrowingLeaf - Model Constraints Demo

GrowingLeaf

Undo Redo Clear Save Load Zoom In Zoom Out Open as SVG Export .leaf Font Size

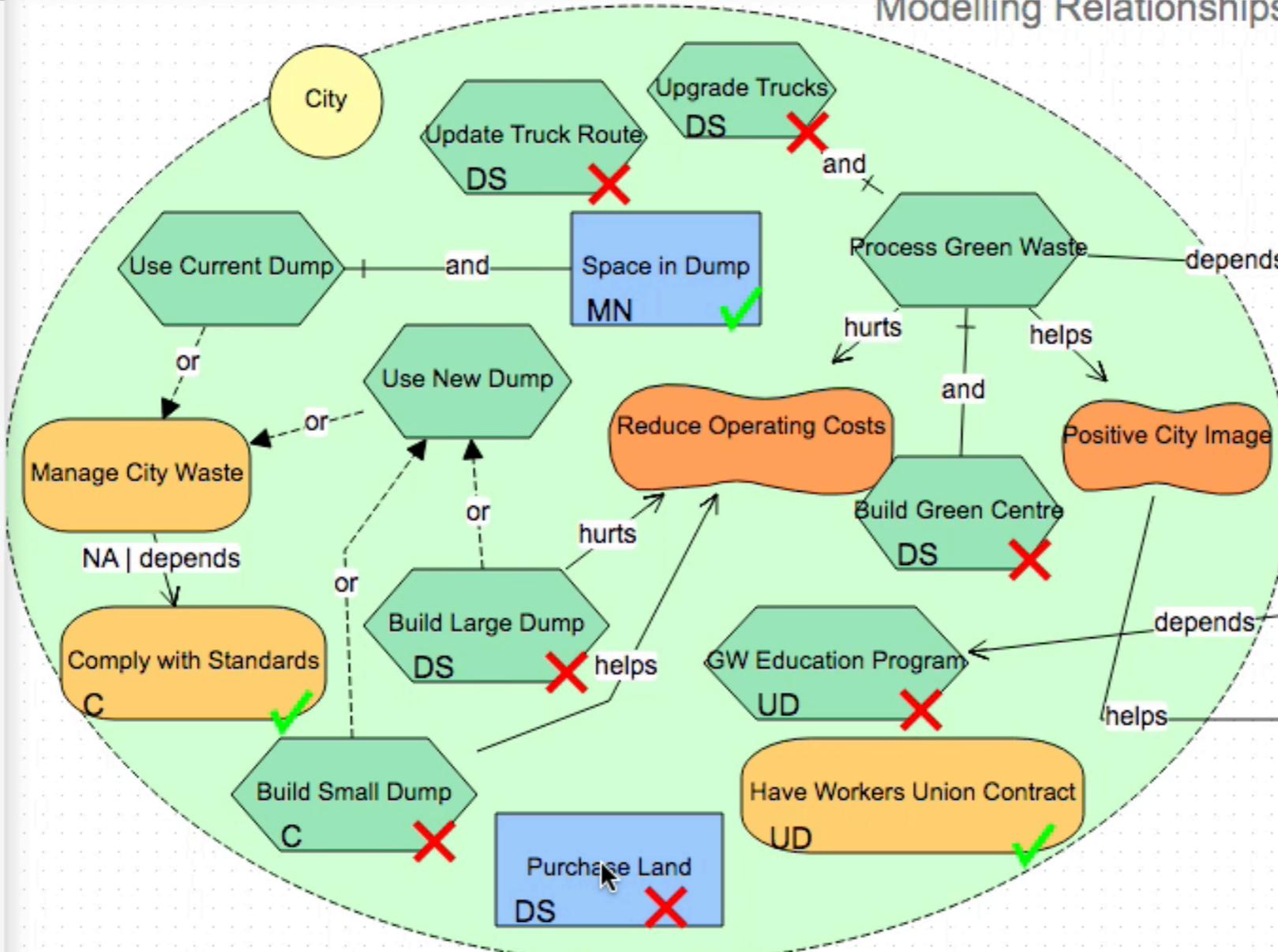
Model Constraints

Analysis

Stencil



Modelling Relationships



Node name:

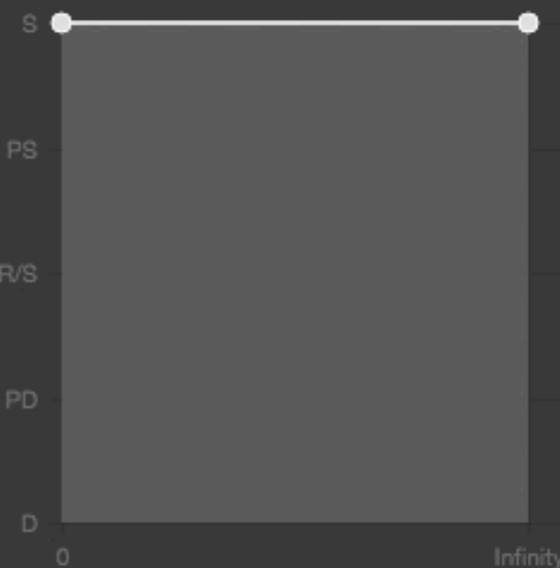
Comply with Standards

Initial Satisfaction Value:

Satisfied

Function Type:

Constant



# GrowingLeaf - Constraints Demo Summary

---

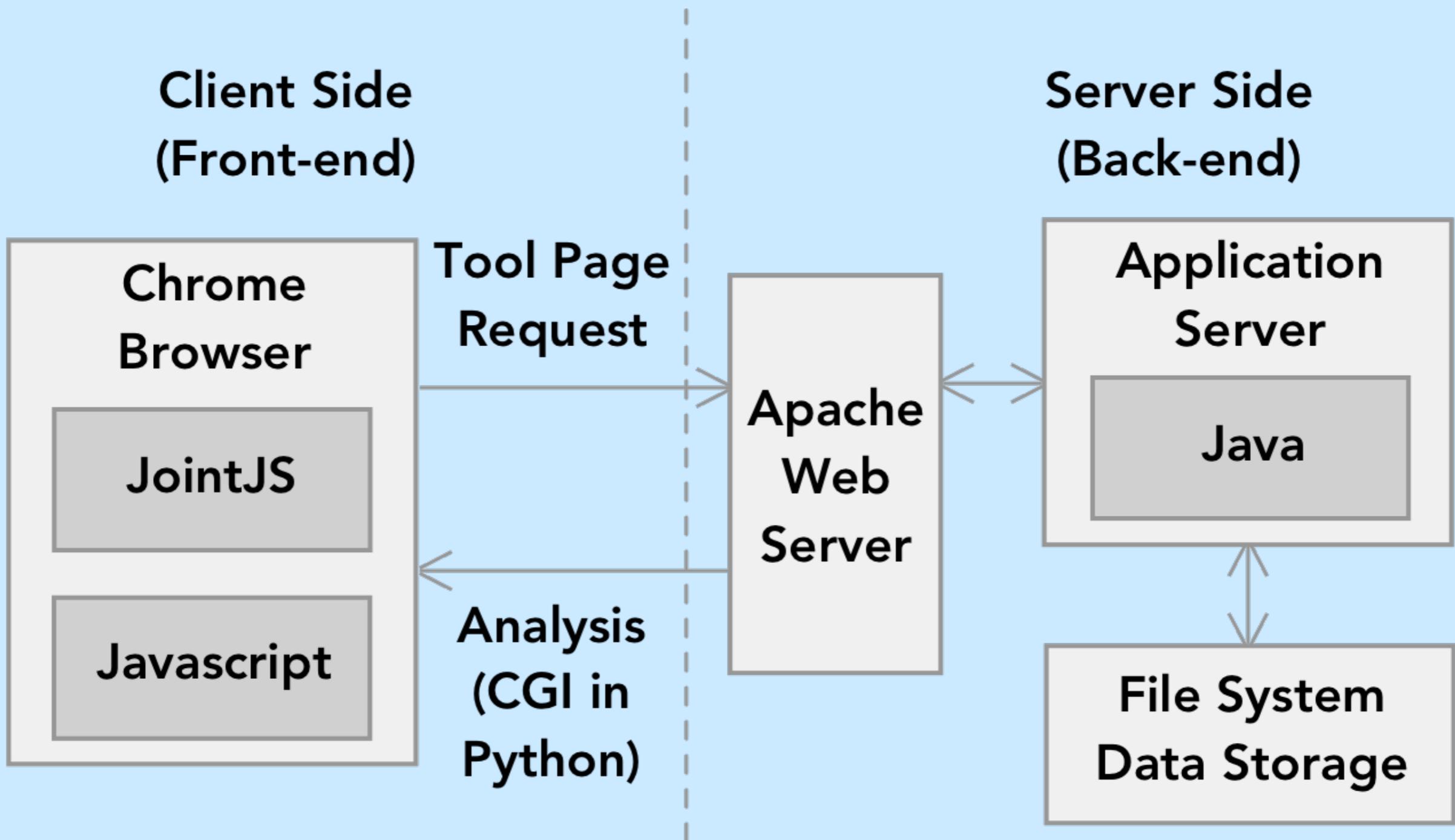
- Adding constraints between EBs

# Outline

---

- Modeling Problem and Tool Justification
- Tool Introduction
- Dynamic Intentions and Analysis
- Tool Functionality
- **Discussion and Validation**
- Status and Future Work

# Architecture



# Design Decisions

---

- Browser versions and updates
- JointJS data model and constrains

# Usability

---

- Two rounds user testing
- Found issues with
  - resizing
  - 'enter' key
  - 'backspace'/'delete' key
  - selecting analysis techniques
- Further user studies are ongoing
- Built several models and examples

# Examples and Case Studies

---

- City transportation planning
- Network maintenance
- Software supply chains
- Technical debt
- Compliance
- Sustainability

Further case studies are ongoing.....

# Ongoing Validation

---

- Evaluate usability / effectiveness with controlled experiment
- Prototype study at this week at iStar and RE
- Please Participate!!

<http://www.cs.toronto.edu/~amgrubb/restudy.htm>

# Where do I get the tool?

---

<http://www.cs.toronto.edu/~amgrubb/growing-leaf>

# Where do I get the tool?

## GrowingLeaf

Click [here](#) to be redirected to a live version of the tool.

<http://www.g-leaf.com>

The screenshot displays the GrowingLeaf software interface. At the top, there is a menu bar with options: Undo, Redo, Clear, Save, Load, Zoom In, Zoom Out, Open as SVG, Export .leaf, Font Size, Model Constraints, and Analysis. Below the menu is a 'Stencil' panel on the left containing icons for Goal (orange rounded rectangle), Task (green hexagon), Soft Goal (orange rounded rectangle), Resource (blue rectangle), and Actor (yellow circle). The main workspace, titled 'Modelling Relationships', shows a goal model diagram. The diagram includes an actor 'Jake' (yellow circle) connected to a goal 'Have Lunch' (orange rounded rectangle). 'Have Lunch' is connected to 'Have Sandwich' (orange rounded rectangle) via an 'or' relationship. 'Have Sandwich' is connected to 'Buy Bread MN' (green hexagon) and 'Prepare Sandwich DS' (green hexagon) via 'and' relationships. 'Buy Bread MN' is connected to 'Have Meat' (orange rounded rectangle) via an 'and' relationship. 'Have Meat' is connected to 'Buy Chicken DS' (green hexagon) and 'Buy Beef DS' (green hexagon) via 'or' relationships. 'Buy Pizza DS' (green hexagon) is connected to 'Have Lunch' via an 'or' relationship. 'Buy Pizza DS' is connected to 'Quick Lunch' (orange rounded rectangle) via a 'helps' relationship. 'Prepare Sandwich DS' is connected to 'Quick Lunch' via a 'hurts' relationship. 'Quick Lunch' is connected to 'Minimal Cost' (orange rounded rectangle) via a 'breaks' relationship. 'Buy Chicken DS' is connected to 'Minimal Cost' via a 'helps' relationship. 'Buy Beef DS' is connected to 'Minimal Cost' via a 'hurts' relationship. Some goal nodes (Buy Bread MN, Buy Pizza DS, Buy Chicken DS, Buy Beef DS) have a green checkmark or a red X. A control panel on the right shows 'Node name: Buy Bread', 'Initial Satisfaction Value: Satisfied', and 'Function Type: Monotonic Negative' and 'Denied'. Below the control panel is a graph showing a function curve that starts at point B on the y-axis and decreases to point A on the x-axis, then remains constant at A up to infinity.

[g-leaf](http://www.g-leaf.com)

GrowingLeaf: is an iStar modeling and analysis tool focused on understanding model evolution and how the evaluations of intentional elements change over time. GrowingLeaf was developed as an extension to [Leaf \(beta\)](#).

# Where do I get the tool?

## GrowingLeaf

Click [here](#) to be redirected to a live version of the tool.

The screenshot displays the GrowingLeaf software interface. At the top, there is a menu bar with options: Undo, Redo, Clear, Save, Load, Zoom In, Zoom Out, Open as SVG, Export .leaf, Font Size, Model Constraints, and Analysis. Below the menu bar is a 'Stencil' panel on the left containing icons for Goal (yellow rounded rectangle), Task (green hexagon), Soft Goal (orange rounded rectangle), Resource (blue rectangle), and Actor (yellow circle). The main workspace, titled 'Modelling Relationships', shows a goal model diagram for an actor named 'Jake'. The diagram includes goals like 'Have Lunch', 'Have Sandwich', 'Have Meat', and 'Minimal Cost', along with tasks like 'Buy Bread', 'Prepare Sandwich', 'Buy Pizza', 'Buy Chicken', and 'Buy Beef'. Relationships between these elements are shown with arrows and labels such as 'and', 'or', 'helps', 'hurts', and 'breaks'. Some task nodes are marked with a red 'X', indicating they are denied. On the right side, a control panel for the selected 'Buy Bread' node shows its name, initial satisfaction value (set to 'Satisfied'), and function type (set to 'Montonic Negative'). Below the control panel is a graph showing a piecewise linear function with points labeled B, PB, R/B, PD, D, A, and Infinity.

<http://w>

[g-leaf](http://w)

GrowingLeaf: is an iStar modeling and analysis tool focused on understanding model evolution and how the evaluations of intentional elements change over time. GrowingLeaf was developed as an extension to [Leaf \(beta\)](#).

# Where do I get the tool?

The screenshot shows a web browser window with the URL [www.cs.toronto.edu/~amgrubb/leaf-ui/Tool.html](http://www.cs.toronto.edu/~amgrubb/leaf-ui/Tool.html). The application interface includes a top menu bar with the following options: Undo, Redo, Clear, Save, Load, Zoom In, Zoom Out, Open as SVG, Export .leaf, Font Size, Model Constraints (highlighted in blue), and Analysis (highlighted in green). On the left side, there is a 'Stencil' panel containing four diagram elements: a yellow rounded rectangle labeled 'Goal', a green hexagon labeled 'Task', an orange rounded rectangle labeled 'Soft Goal', and a blue rectangle labeled 'Resource'. Below these is a larger diagram element consisting of a yellow circle labeled 'Actor' overlapping a larger green circle with a dashed border. The main workspace is a large grid titled 'Modelling Relationships'. At the bottom left, there is a copyright notice for the University of Toronto and a 'Powered by:' section for client IO and JointJS.

Stencil

- Goal
- Task
- Soft Goal
- Resource

Actor

Modelling Relationships

Copyright 2015-2016.  
University of Toronto  
Department of Computer Science.  
All rights reserved.

Powered by:

Copyright 2014-2016.  
client IO. All rights reserved.  
JointJS: an HTML 5 diagramming  
component.  
<http://jointjs.com>

# Where do I get the tool?

The image shows a screenshot of a web browser displaying the GrowingLeaf application. The browser's address bar shows the URL [www.cs.toronto.edu/~amgrubb/leaf-ui/Tool.html](http://www.cs.toronto.edu/~amgrubb/leaf-ui/Tool.html). The application's interface includes a top navigation bar with the following menu items: Undo, Redo, Clear, Save, Load, Zoom In, Zoom Out, Open as SVG, Export .leaf, Font Size, Model Constraints (highlighted in blue), and Analysis (highlighted in green). On the left side, there is a 'Stencil' panel containing four diagram elements: a yellow rounded rectangle labeled 'Goal', a green hexagon labeled 'Task', an orange rounded rectangle labeled 'Soft Goal', and a blue rectangle labeled 'Resource'. Below these is a larger diagram element consisting of a yellow circle labeled 'Actor' overlapping a larger green circle with a dashed border. The main workspace is a large grid with the title 'Modelling Relationships'. In the center of the grid, the text 'Use Google Chrome' is displayed. At the bottom left, there is a copyright notice for the University of Toronto and a logo for JointJS, with the text 'Powered by: Copyright 2014-2016. client IO. All rights reserved. JointJS: an HTML 5 diagramming component. <http://jointjs.com>

# Where do I get the tool?

---

<http://www.cs.toronto.edu/~amgrubb/growing-leaf>

Join the development team.

# Future Work

---

- Update tool to use iStar 2.0 Language Guide
- External industrial case study
- Improve server connection (security)
- Multiple users to simultaneously edit
- Development for other browsers

# Questions?

# GrowingLeaf: Supporting Requirements Evolution over Time

---

GrowingLeaf

<http://www.cs.toronto.edu/~amgrubb/growing-leaf>

Tool Study at RE'16:

<http://www.cs.toronto.edu/~amgrubb/restudy.htm>