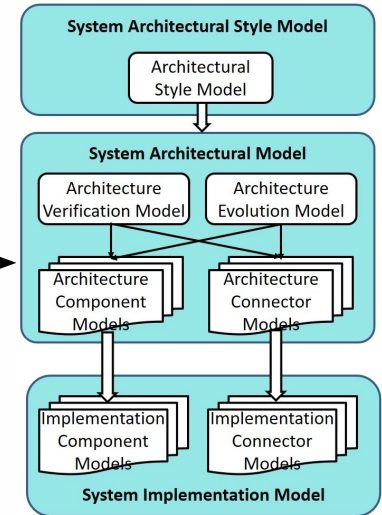
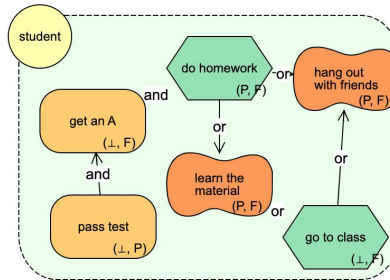


An Assistive Approach for Learning Goal Modeling

Karenn Kung and Alicia M. Grubb

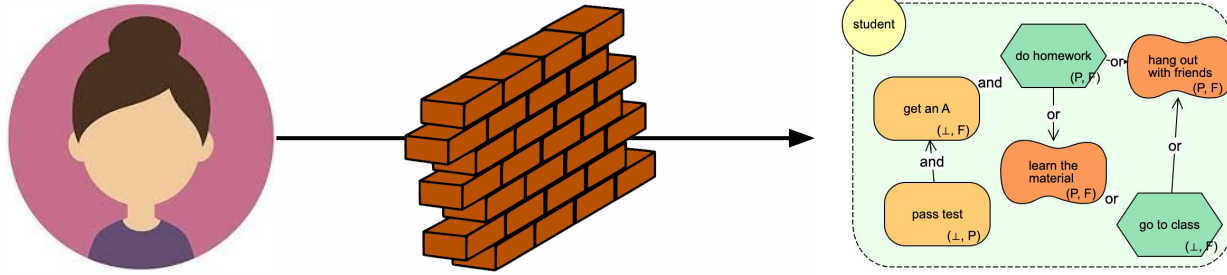
Smith College

Context



Wu, Di & Ban, Xiaojuan & Wang, Hao. (2017). COSE: A Composable Ocean Simulation Environment.

Problem



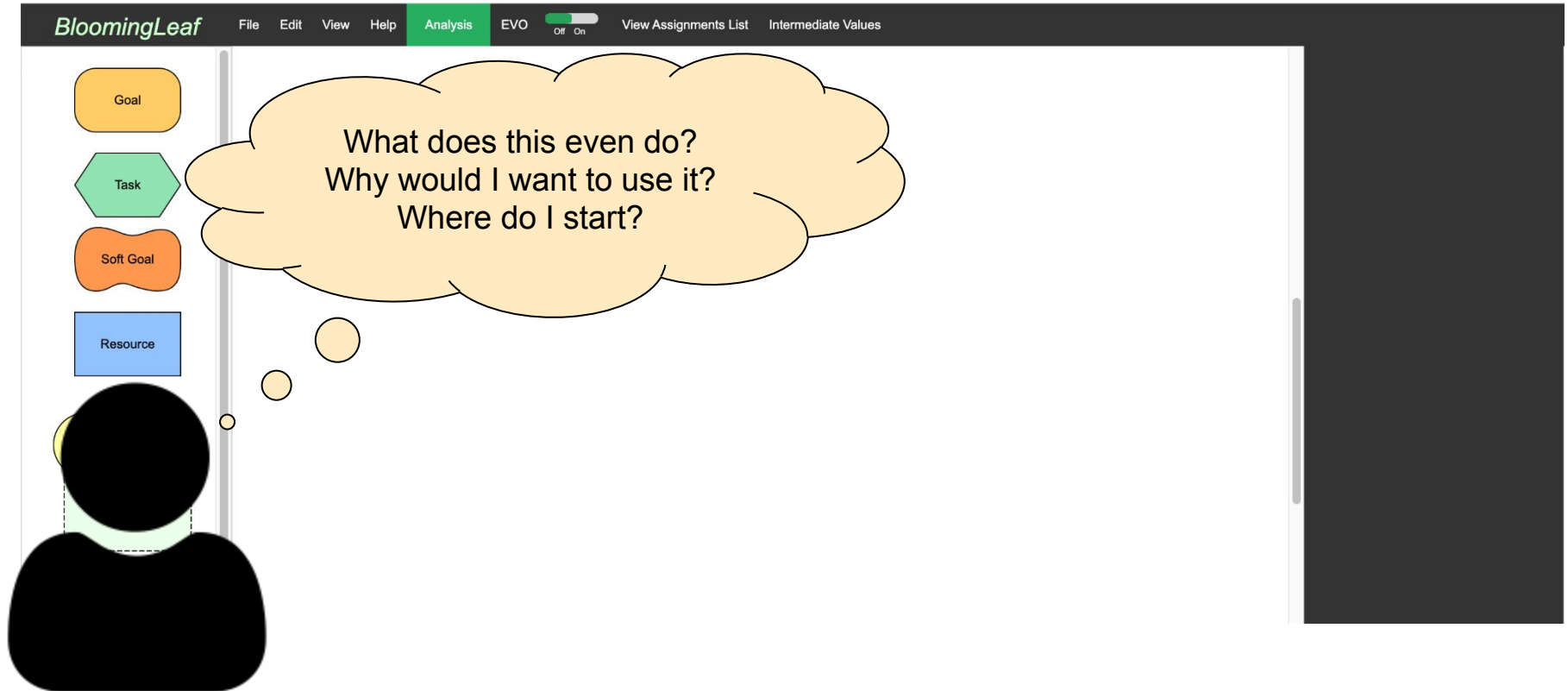
stakeholder

- Complexity of tooling
- Lack of available training
- Difficulty understanding models

Previous State

The screenshot displays the BloomingLeaf software interface. The top navigation bar is dark grey and contains the following elements from left to right: the logo 'BloomingLeaf' in a stylized font, a menu with 'File', 'Edit', 'View', and 'Help', the 'Analysis' tab highlighted in green, the 'EVO' section with a green slider set to 'On' (labeled 'Off' and 'On'), and two links: 'View Assignments List' and 'Intermediate Values'. On the left side, there is a vertical toolbar with five icons: an orange rounded rectangle labeled 'Goal', a green hexagon labeled 'Task', an orange rounded rectangle labeled 'Soft Goal', a blue rectangle labeled 'Resource', and a yellow circle labeled 'Actor' next to a light green dashed rectangle. The main workspace is a large white area. On the right side, there is a dark grey vertical panel with a scrollbar.

Previous State



Previous State

BloomingLeaf File Edit View Help Analysis EVO Off On View Assignments List Intermediate Values

Goal
Task
Soft Goal
Resource

What does this even do?
Why would I want to use it?
Where do I start?

Absolute and Relative Assignments

Max Absolute Time
[0] [100]
Absolute Time Points
[0] [100]

Relative Intention Assignments

Epoch Boundary Name 1 Relationship Epoch Boundary Name 2

Absolute Intention Assignments

Epoch Boundary Name	Function	Assigned Time	Action
untitled : A	RC		Unassign
untitled : A	RC		Unassign
untitled : A	UD	3	Unassign
untitled : B	UD	10	Unassign
untitled : C	UD		Unassign

Absolute Relationship Assignment

Link Type	Source Node name	Dest Node name	Assigned Time	Action
-----------	------------------	----------------	---------------	--------

Presence Condition Assignments

Element	Type	Available Interval
---------	------	--------------------

No Relationship
✓ And-Decomposition
Or-Decomposition
++
--
+
-
+S
++S
-S
--S
+D
++D
-D
--D
Not Both (None)
Not Both (Denied)

Absolute Intention Assignments

Epoch Boundary Name	Function	Assigned Time	Action
untitled : A	RC		Unassign
untitled : A	RC		Unassign
untitled : A	UD	3	Unassign
untitled : B	UD	10	Unassign
untitled : C	UD		Unassign

Node Name:
untitled

Initial Satisfaction Value:
[no value]

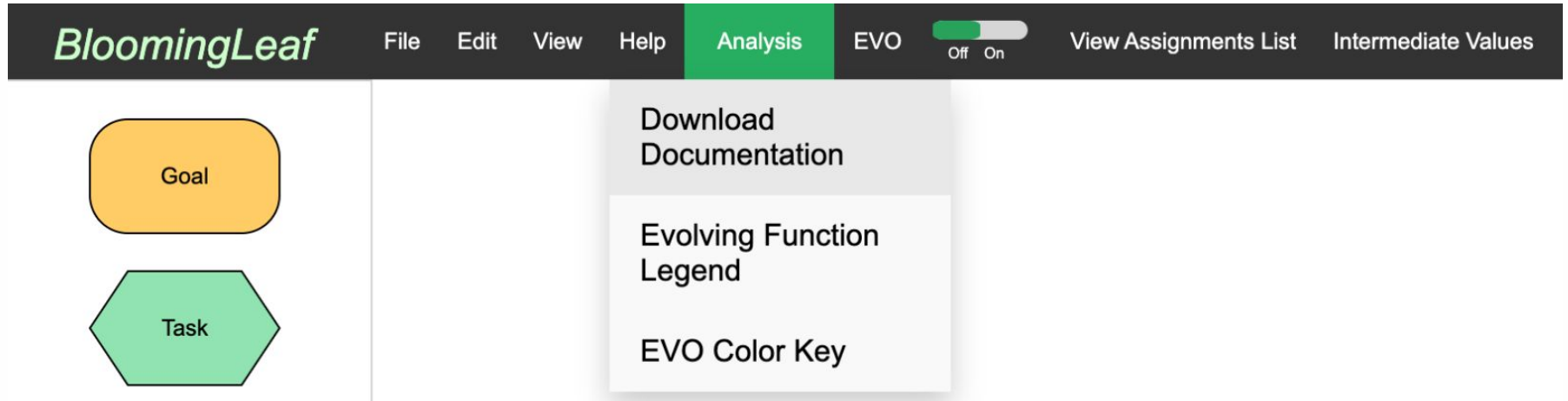
Function Type:
User Defined

Epoch Time
0 0 Constant Detailed [0, 100]

Intervals

Available:
0 - 100
Flip Interval

Previous State



Previous State

Introduction to Goal Modeling in TROPOS

1 / 12 | - 100% + | [Icons]

1

Goal Modeling and Blooming Leaf Training

2

Domain Model

3

Function Types

Goal Modeling and Blooming Leaf Training

Intentions and Actors:

Goal

Soft Goal

Actor

Task

Resource

Goal models consist of actors and intentions

Types of intentions are goals, soft goals, tasks, and resources

Tropos Evidence Pairs:

Fully Satisfied

Partially Satisfied

None

Partially Denied

Fully Denied

Conflicting Evidence Pairs

Other Tools

IBM Rhapsody

Basic systems engineering design in Rhapsody®

Last Updated: 2025-06-06

This tutorial demonstrates how to apply a SysML profile to project, and how to design a basic structure and behavior.

Learning objectives

The tutorial demonstrates the following key concepts:

- Relationships of requirements to use cases
- Methods to define behavior
- Methods to define structure
- Relationships of required behavior to system architecture and validation approaches
- Techniques for handing off the project to software developers

Note: The tutorial begins with a SysML model that contains some artifacts. You can use the *Spa and pool temperature control architecture* model to start working with this tutorial.

Time required

4 hours

– **Introduction: Basic systems engineering design in IBM Engineering Systems Design Rhapsody**

The systems engineering tutorial starts with a SysML project containing artifacts for an outdoor spa pool temperature controller. Instructions and demonstrations help you to complete the simple architecture and hand it off to software engineers.

– **Tutorial setup: Downloading the starting point project**

In this lesson, you download the starting point project for the tutorial and open it.

– **Lesson 1: Analyzing requirements**

In this lesson, you view a short tour of the requirements and requirements table and edit the project.

– **Lesson 2: Tracing requirements in use cases**

In the previous lesson, you reviewed the requirements and learned the difference between functional and nonfunctional requirements. This lesson contains a short video showing the use cases in the starting point project and their relationships. Analysis of the use cases gives you information about both the structure and the behavior of the system.

– **Lesson 3: Creating standard value types**

Most of the attributes have already been added to the SystemUnderControl block. In this lesson, you add the mass of the water to the block. Since the mass of water is expressed in kg, you begin by specifying an applicable

Catia MagicDraw

User Guide

Welcome! This user guide will walk you through the basics of using the modeling tool, including working with projects, setting up the modeling environment, collaborating with the help of [Teamwork Cloud](#), generating reports, and more.

Use the search box to find a specific topic or select one from the list below:

- Getting started
- Working with projects
- Authentication with 3DEXPERIENCE platform
- CATIA applications
- Diagrams
- Diagramming
- UML elements
- Auxiliary diagram symbols
- Working with model elements
- Tools
- Model analysis
- Collaborative modeling
- Report Wizard
- Exchanging data between tables and Excel or CSV files

MATLAB Simulink



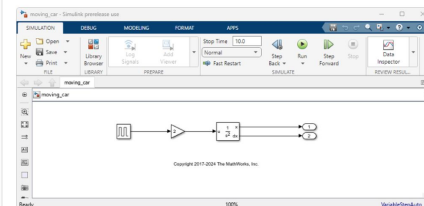
Create a Simple Model

R2025a

You can use Simulink® to model a system and then simulate the dynamic behavior of that system. The basic techniques you use to create a simple model in this tutorial are the same as those you use for more complex models. This example simulates simplified motion of a car. A car is typically in motion while the gas pedal is pressed. After the pedal is released, the car coasts and comes to a stop.

A Simulink block is a model element that defines a mathematical relationship between its input and output. To create this simple model, you need four Simulink blocks.

Block Name	Block Purpose	Model Purpose
Pulse Generator	Generate an input signal for the model	Represent the accelerator pedal
Gain	Multiply the input signal by a constant value	Calculate how pressing the accelerator affects the car acceleration
Second-Order Integrator	Integrate the input signal twice	Obtain position from acceleration
Output	Designate a signal as an output from the model	Designate the position as an output from the model



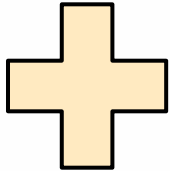
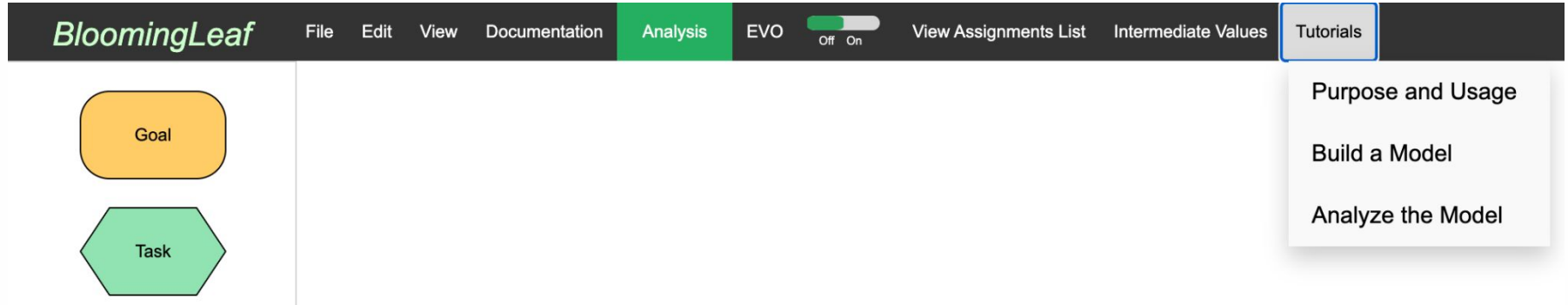
Simulating this model integrates a brief pulse twice to get a ramp. The input pulse represents a press of the gas pedal. 1 when the pedal is pressed and 0 when it is not. The output ramp is the increasing distance from the starting point.

Open New Model

Use the [Simulink Editor](#) to build your models.

1. Start MATLAB®. From the MATLAB toolbar, click the [Simulink](#) button.

Proposed Solution



Easily accessible tutorials embedded in the tool
Navigable step-by-step or as-needed
Including development of example model

Tutorial Content

Uncertainties reported by novice user

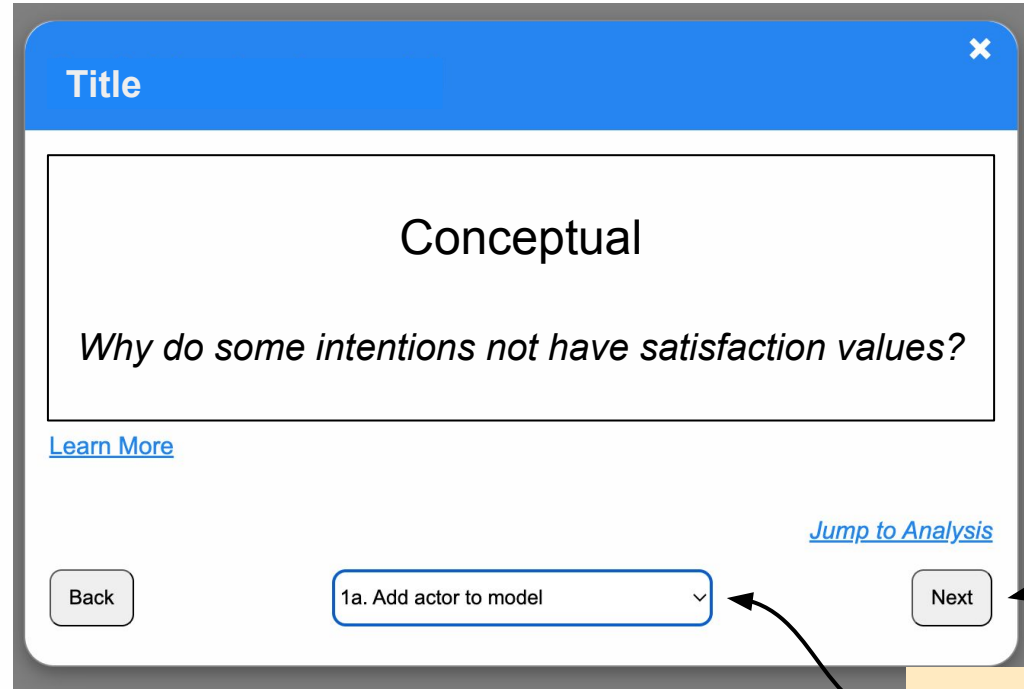
Novice user example questions:

	Mechanical	Conceptual
General Modeling:		What is the point of BloomingLeaf?
Creating:	Link button is not intuitive	Why do some intentions not have satisfaction values?
Analyzing:	When do EVO colors show up?	How to decide number of time points, what time frame do time points symbolize

Tool Structure - Build

Mechanical

Link button is not intuitive

A mockup of a software tool interface. It features a blue header bar with the text 'Title' and a close button (X). Below the header is a large white box containing the word 'Conceptual' and the question 'Why do some intentions not have satisfaction values?'. Underneath this box is a blue link labeled 'Learn More'. At the bottom of the interface, there is a 'Back' button on the left, a dropdown menu in the center with the text '1a. Add actor to model' and a downward arrow, and a 'Next' button on the right. A blue link labeled 'Jump to Analysis' is positioned above the 'Next' button.

Step-by-step

As-needed

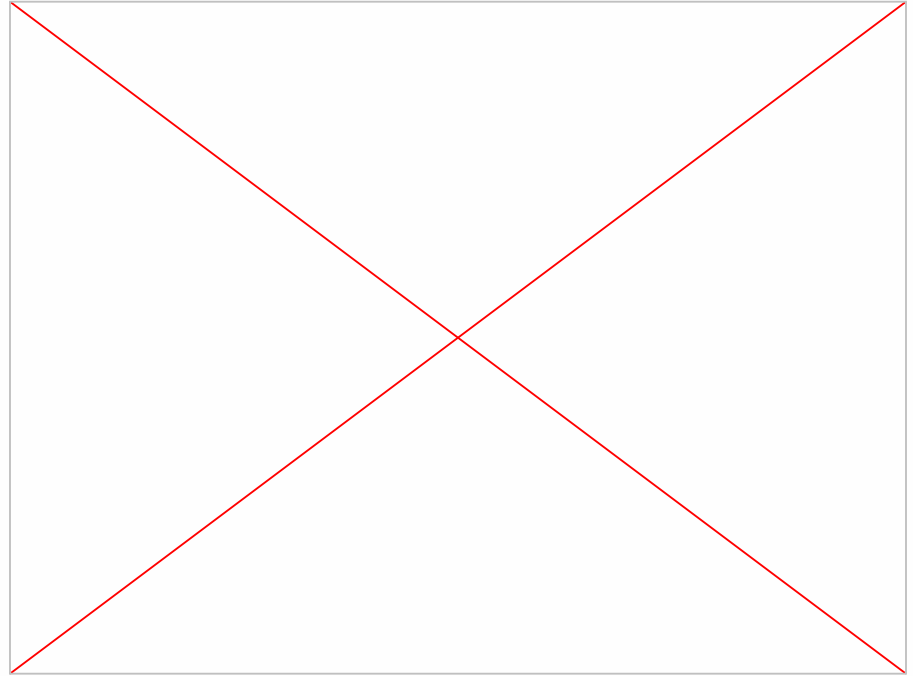
Tool Structure - Build

1. Create the model

- 1a. Add actor to model
- 1b. Use actor inspector
- 1c. Add intentions to the model
- 1d. Use intention inspector
- 1e. Link intentions

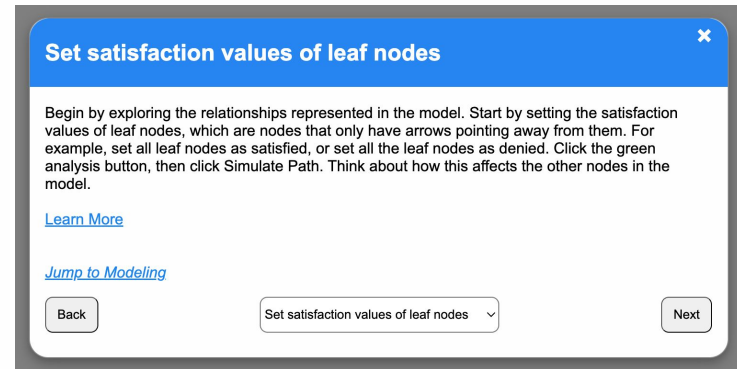
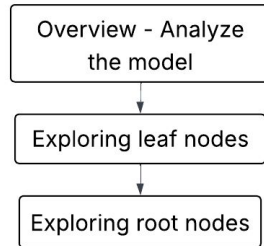
2. Adding evolutionary information

- 2a. Set initial satisfaction value
- 2b. Set evolving functions
- 2c. Limit presence intervals
- 2d. Change max absolute time
- 2e. Set absolute time points
- 2f. Set relative intention assignments
- 2g. Set absolute relationship assignments
- 2h. Change presence intervals
- 2i. Use intermediate values table



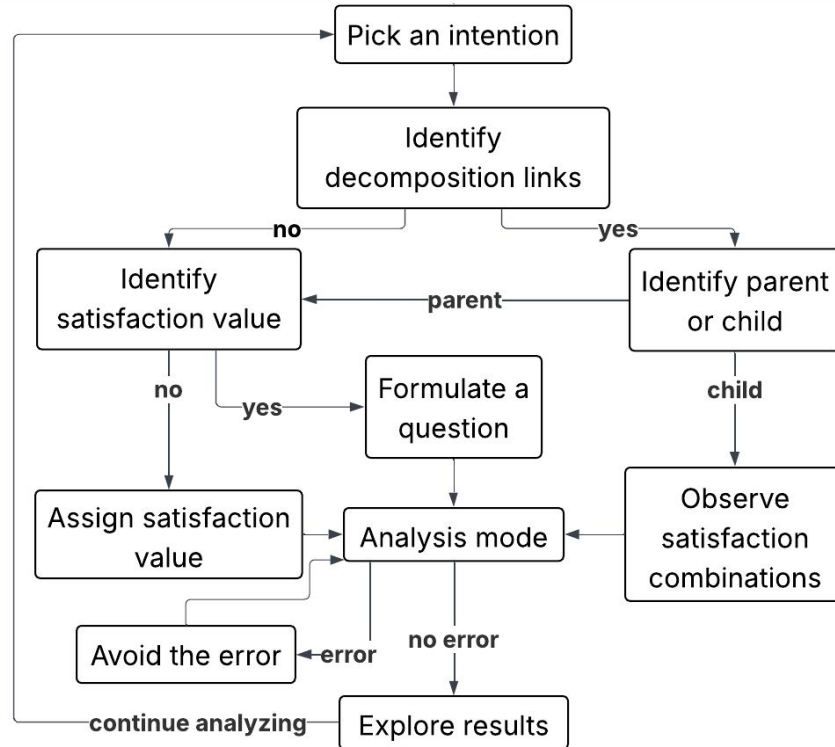
Tool Structure - Analyze

1



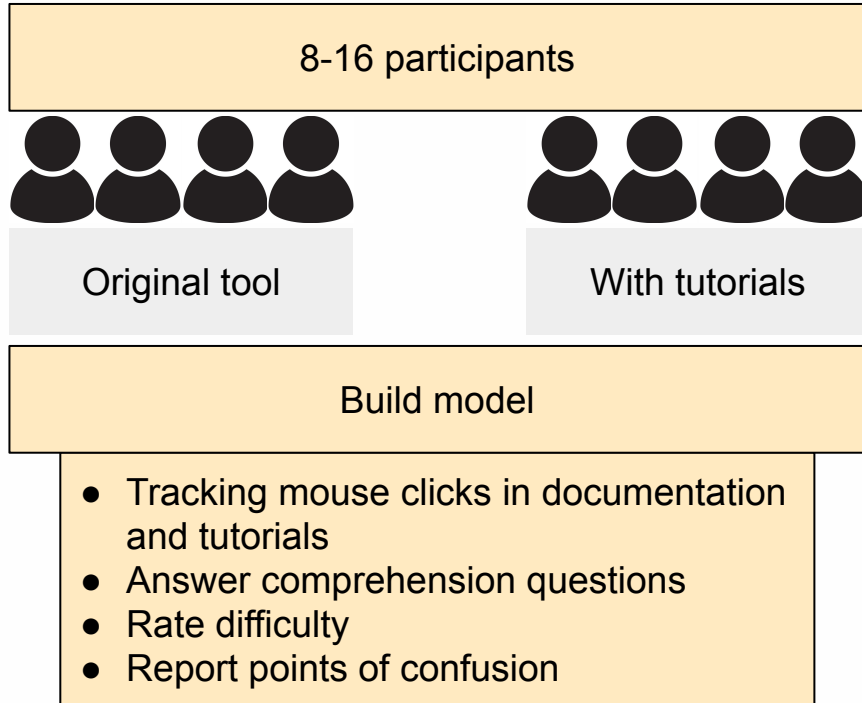
Tool Structure - Analyze

2



The screenshot shows a web-based tool titled 'Identify parent or child'. It asks the user 'Is this intention the child or parent within the or-link?'. There are two buttons: 'Child' and 'Parent'. A dropdown menu is currently set to 'Identify parent or child'. There are also links for 'Learn More' and 'Jump to Modeling'.

Plan for Validation



Thank You

Thank you to Molly Daniel and Christine Dong for their contributions to this project.

This material is based upon work supported by the National Science Foundation under Award No. 2104732.