Stimulus Tutorial

Quick Start Guide

Jean-Michel Bruel, bruel@irit.fr

Version 1.0, 2018-11-07

Table of Contents

In this document you will find:

- the explanation of its context in Section 1,
- the detailed explanation of the case study which starts in Section 2,
- some explanations on some basic features of Stimulus in Section 5, and
- some explanations on some advanced features of Stimulus in Section 6.

A **F**requently **A**sked **Q**uestions (FAQ) is also available in Section 7.

1. Context

1.1. About this tutorial

This tutorial is based on the official one from Argosim and available here: https://download.argosim.com/index.php/s/5ZszF09tl0rd4gv/download.

1.2. Installation and requirements

This tutorial use the version `2018.09.1` of Stimulus.

1. Launch the Stimulus installer
2. Generate the number required for the licence
3. Send the email as required in order to get the licence file

The tool requires Windows! And no virtual machines allowed.

2. Case study

2.1. Description

This exercice provides a short introduction to Stimulus. It explains how to create a new project and define your first system specification. You will also learn how to write and simulate basic requirements, and discover the main debugging features of the tool.

3. First steps

3.1. Launch Stimulus

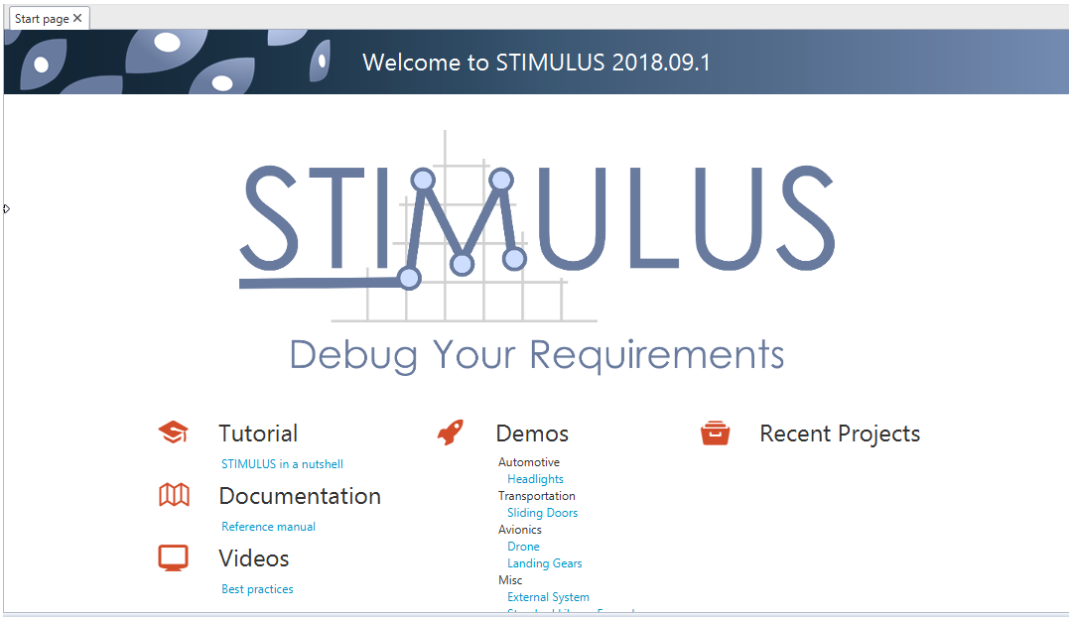

Figure 1. Stimulus start page

3.2. Create a new `project`

The first step is to create a new project. To do so, Click the **[New project]** located in the toolbar or open the **File** and select `New Project`. A pop-up window appears to configure basic project properties, as shown in Figure 2.

Name your project `QuickStartGuide`, for example.

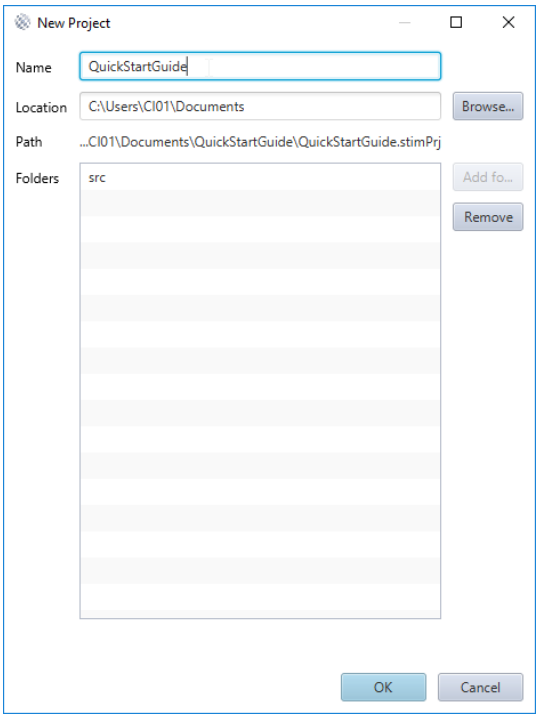

Figure 2. New project

You can set the project name in the `Name` field, and define your working directory in the `Location` field. Then click **[OK]** to finish. Your project now appears in the `Projects` tab as shown in Figure 3. The project tree will allow you to manage your project models just like a file system would do.

Figure 3. New project in the `Projects` tab

3.3. Create a new package

Projects can be organised hierarchically with `packages`, that contain editable model elements such as requirements, glossaries, etc. To create a package, select the parent directory or package in your project tree (the `src` forlder), then open the main **File › New › Package** as shown in Figure 4.

As a rule in Stimulus, you can also right-click to perform the same action from the contextual menu.

When created, the new package is selected and editable, which allows you to rename it. If you want to rename it later, you need to select it by clicking on it, then clicking again after a short delay to make it editable.

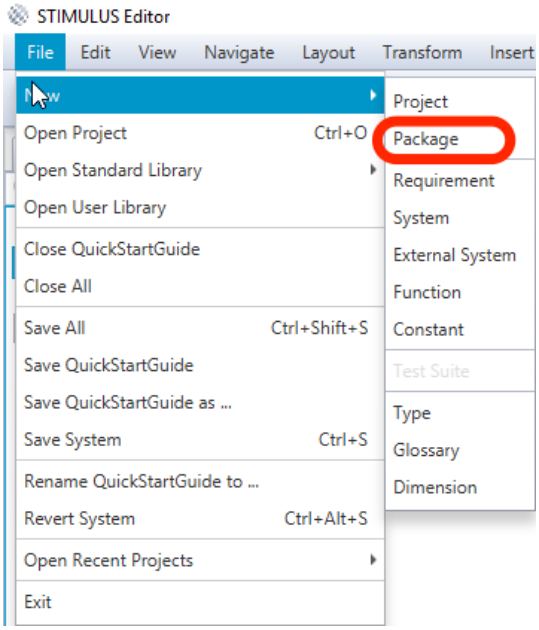

Figure 4. New package

3.4. Create a new `system`

To add a system into the newly created package, select it and choose **File › New › System** in the menu (see Figure 5).

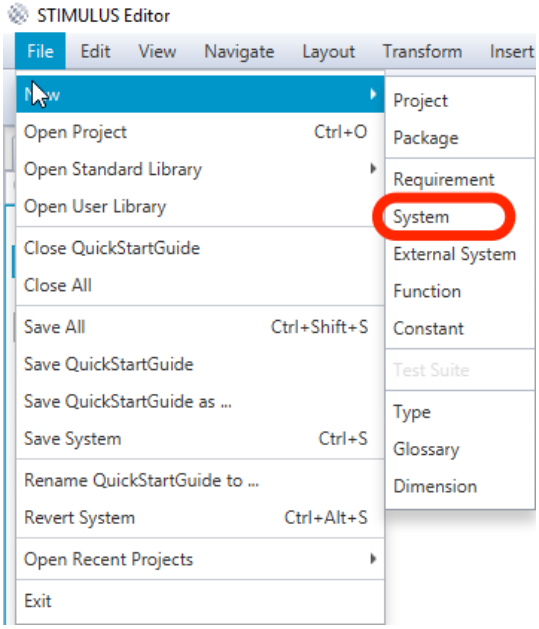

Figure 5. New system

A `system` can be seen as "anything with ports (inputs/outputs)".

3.5. Create a new `port`

Once the system is created, you can define the interface with its environment by declaring a list of named ports (Input and output signals). To declare a port, click the **[plus shaped]** next to the ports label in the `Interface` panel on the right side of the window. A new port appears in the corresponding section that can be renamed as shown in Figure 6.

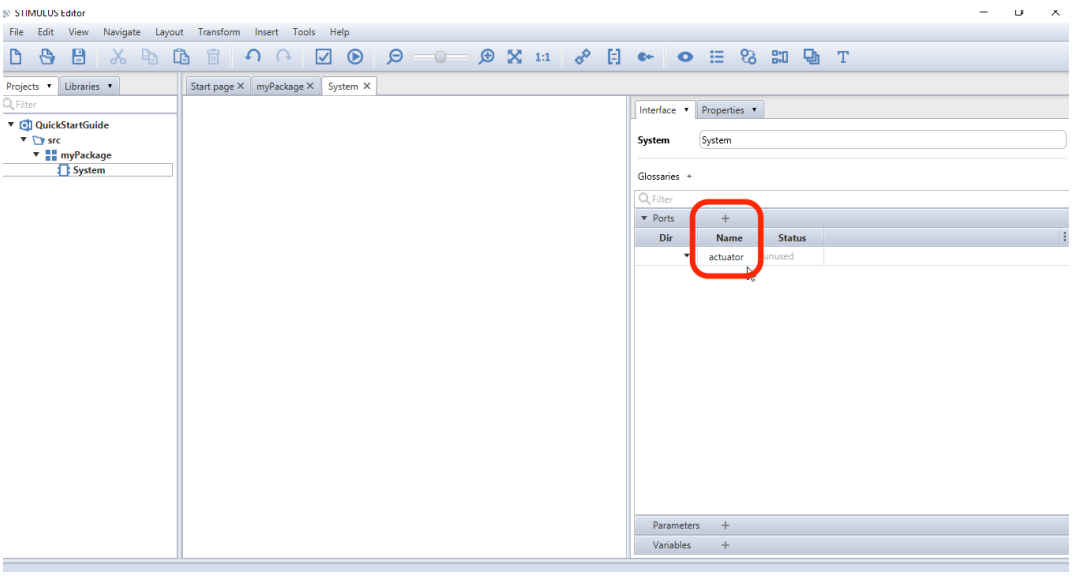

Figure 6. New port

3.6. Perform your first simulation

Your newly created system is ready for simulation, even if it is still empty! To launch the simulation of a selected system tab, press the **[Play]** button (). The Simulation screen appears on the lowest part of the window (see Figure 7).

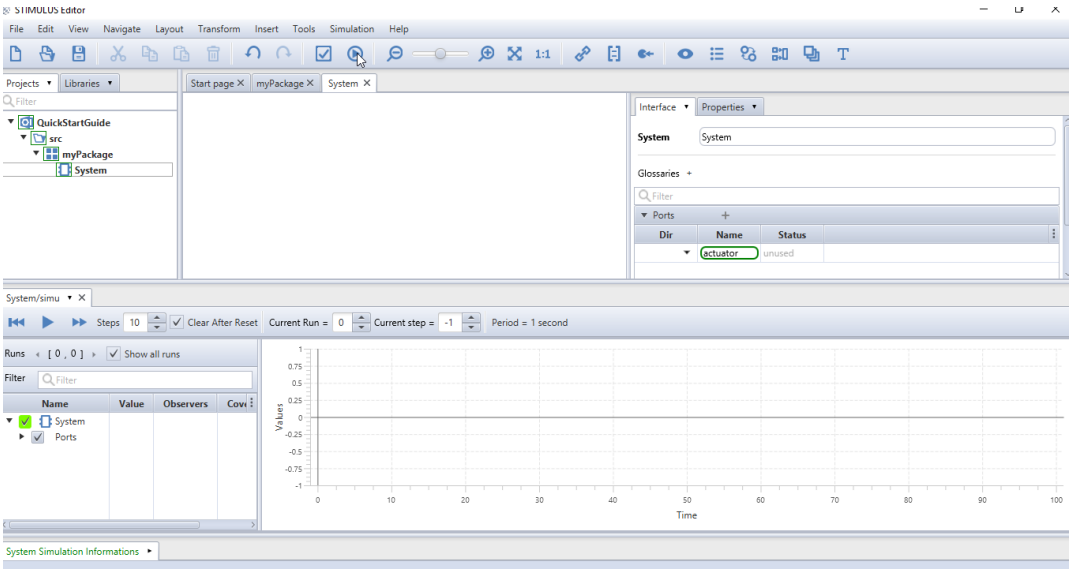

Figure 7. Simulation view

To have a better view on the simulation, you can "detach" the panel from the tool, by right-clicking on the panel and select **Undock**.

3.7. Then play with the simulation pad (Simulate buttons)

You can play with the simulation interface (e.g., Figure 8), or if you want to learn the simulation interface, go to Section 7.2.

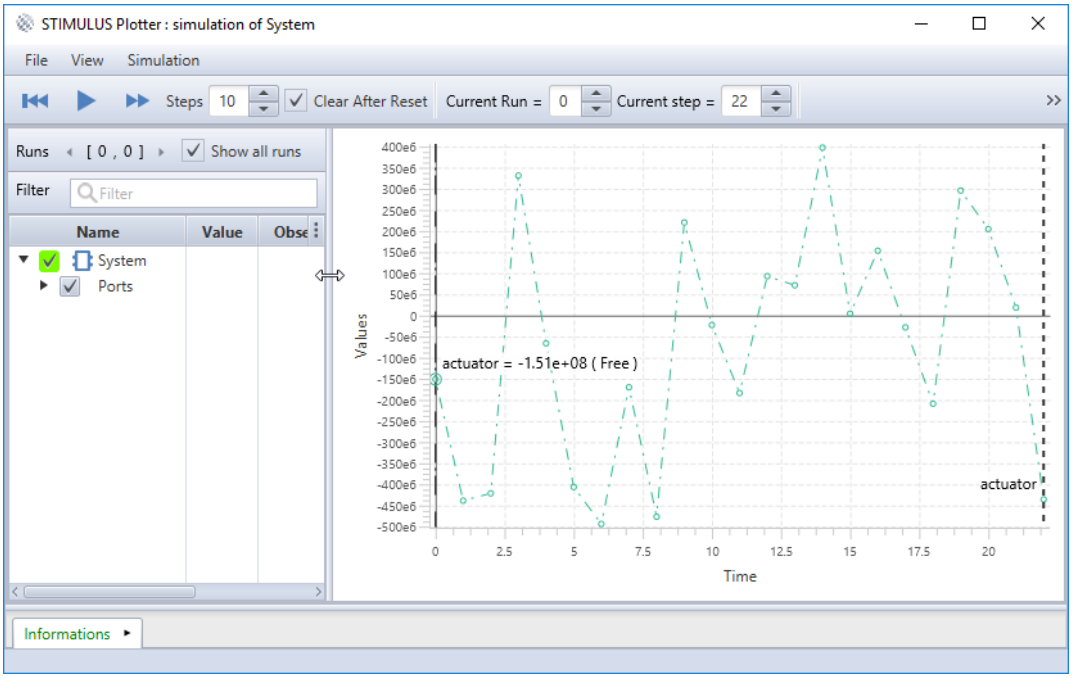

Figure 8. First simulation

Since there is no specified requirement in this system, Stimulus will choose a random value within the range of possible values. Moreover, since no data type has been specified for the port, then Stimulus will assume a numeric type and it will choose a value between minus infinity and plus infinity.

4. Requirements

4.1. First requirement

From here, we can write requirements to describe the behaviour of the system. To help you with this task, a standard library of sentence templates is provided with Stimulus. To display the standard library navigation tree, click on the **[Libraries]** tab on top of the project navigation panel. If the library is not available, open it by choosing **File › Open Standard Library › Stdlib** in the menu. The library is then displayed as shown in Figure 9. All the items of the library are organised into packages.

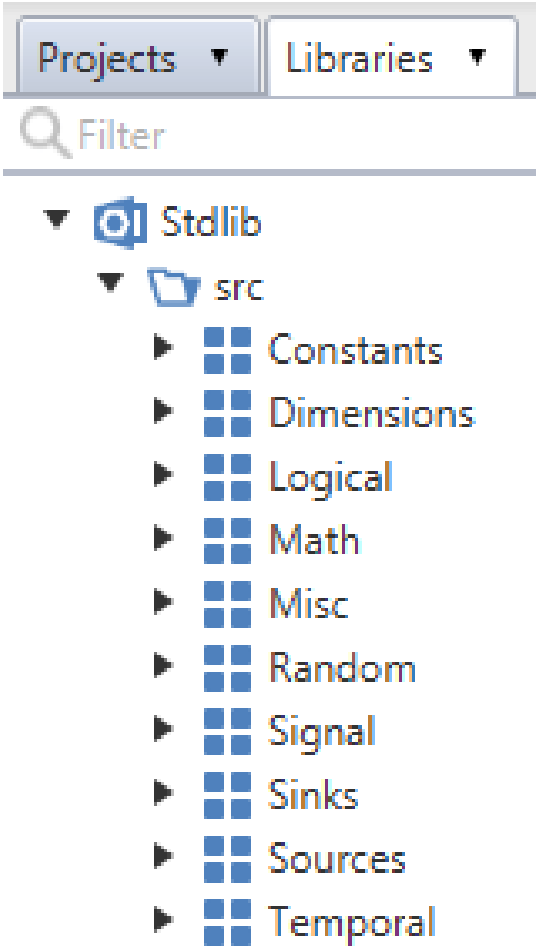

Figure 9. Standard library navigation tree

To build a sentence from a library item, drag and drop it into the system's panel as shown in Figure 10. In this example, we use the **Logical › InRange › #[min;max]** pattern. A sentence with gaps appears in the system's panel.

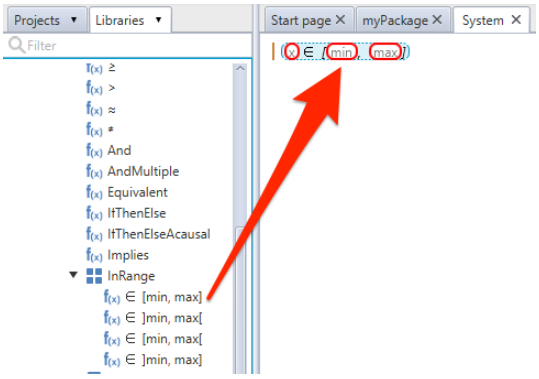

Figure 10. Add a library item to a system

Stimulus allows you to modify its syntax, also known as `Format`, by opening its contextual menu and choosing **Format › x shall be in range...**.

To complete the sentence, we need to fill in the gaps. To do so, drag and drop the port of the system directly into the first gap (see Figure 11).

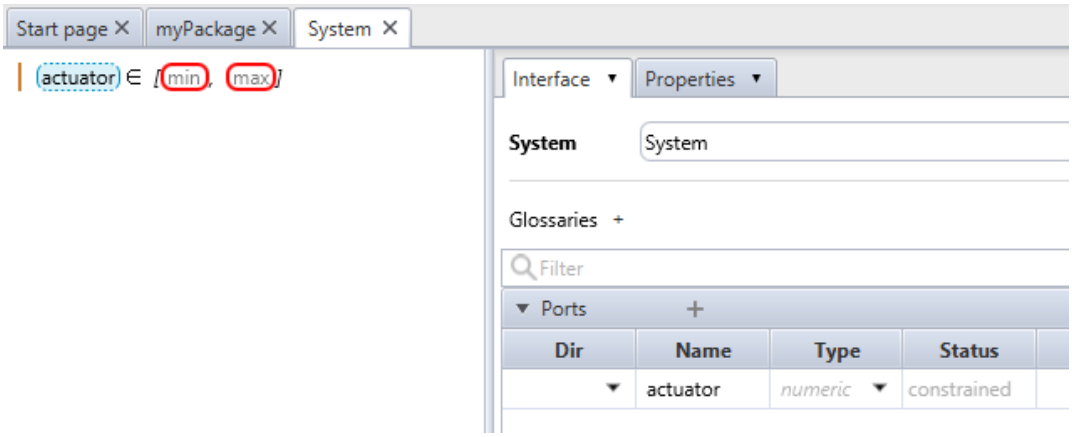

Figure 11. Fill in a gap with a system port

Finally, fill in the remaining gaps to get your first complete requirement.

4.2. First simulation

Start a new simulation by clicking the **[Play]** button ( ). We observe in Figure 12 that all values of `actuator` stay between `0` and `10`, satisfying the requirement.

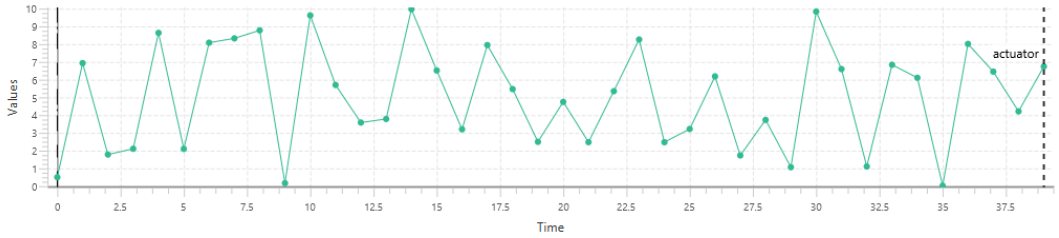

Figure 12. Simulation of the first requirement

Performing another simulation will generate a new behaviour which also satisfies the requirement.

4.3. Real-time requirement & simulation

Stimulus allows you to write real-time requirements, that constrain the system behaviour at some specific time instants or time periods. The `Temporal` library package provides templates to write such requirements. Let us write a new requirement using the **Temporal › When** template as shown in Figure 13. It applies the `<BODY>` constraint when the `condition` is true.

To fill the gaps, open the `Logical` library package, drag and drop the `=` into the `condition` and the `>=` into the `<BODY>` as shown in Figure 13. You can notice that combining sentence templates allows you to write complex requirements.

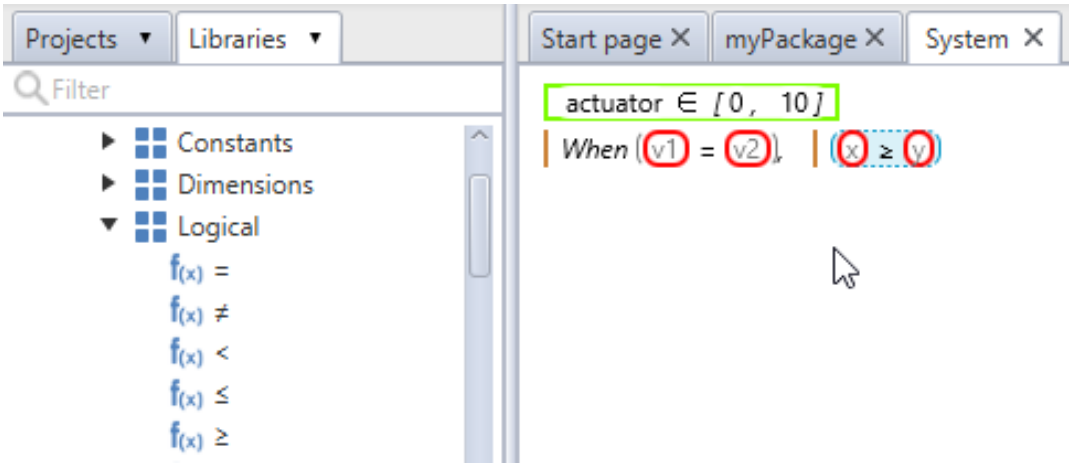

Figure 13. Add a When requirement

Finally, create a new port sensor in the same way as you did for the port actuator. Drag and drop the new port sensor to v1 double click on v2 and type true. Drag and drop the port actuator to x, double click on y and type 5 to complete the requirement with ports and values.

Launch the simulation by clicking the **[Play]** button ( ). The plot window in Figure 14 shows two curves: the green curve for the sensor and the orange one for the actuator. Since Stimulus has inferred that the sensor is a Boolean input and no constraint applies, it generates random values. The orange curve shows the behaviour of the actuator. Depending on the value of the sensor input, the requirement actuator >= 5 will apply or not. When it applies, Stimulus will solve this requirement together with the first one, therefore will generate values between 5 and 10. When sensor is false, it only satisfies the first requirement.

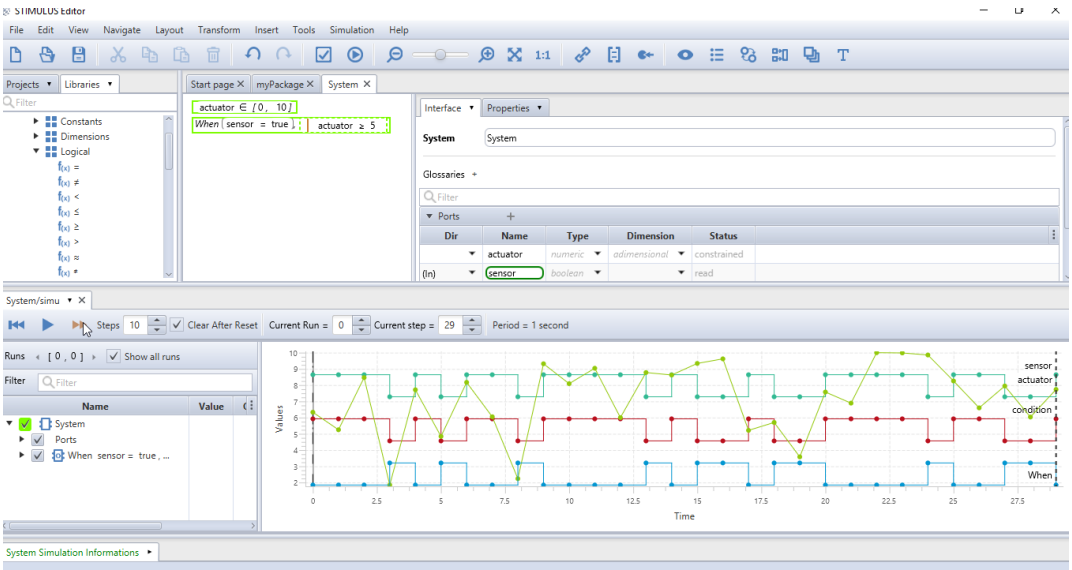

Figure 14. Simulate two real-time requirements

You can select the values shown in the simulation. In Figure 15, the condition has been deselected for example.

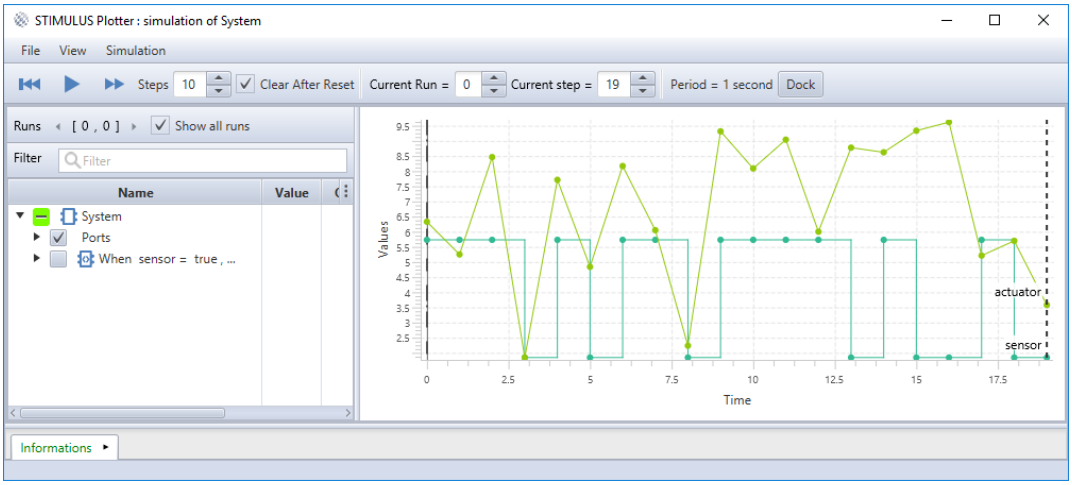

Figure 15. Simulate two real-time requirements

4.4. Analysis of the simulation

In the last section, we have seen how to build requirements and simulate them. Stimulus also provides you more ways to analyse your system requirements during simulation.

To illustrate the highlight feature, let us take a closer look at the simulation of the previous system. In the top part of the window, requirements are highlighted as in Figure 16 to indicate which constraint is active at a given step of the simulation.

Figure 16. Highlighted requirements

The selected step is given by a vertical dotted line as shown in Figure 15. The step number is displayed in the `Current step` text field (16 in this example). You can select the current step either by clicking in the simulation window or by typing a specific step number into the `Current step` text field.

Going back to Figure 16, constraints that are highlighted in solid gray are not active: at the selected step, `sensor` is false, therefore the condition in the `When` statement is not verified and the constraint is not active. As a consequence, the value of `actuator` is still chosen between 0 and 10 but is not forced to be greater than or equal to 5.

Let us see what happens when adding the third requirement as shown in Figure 17.

Figure 17. Requirement using time

To edit it, use the **Temporal › AfterPeriod** and the **Logical ›** < provided in the standard library. At simulation time, the execution stops, a red vertical line appears and an error message is added to the log panel at the bottom of the window, see Figure 18. It shows a so-called conflict, which happens when at least two requirements are contradictory.

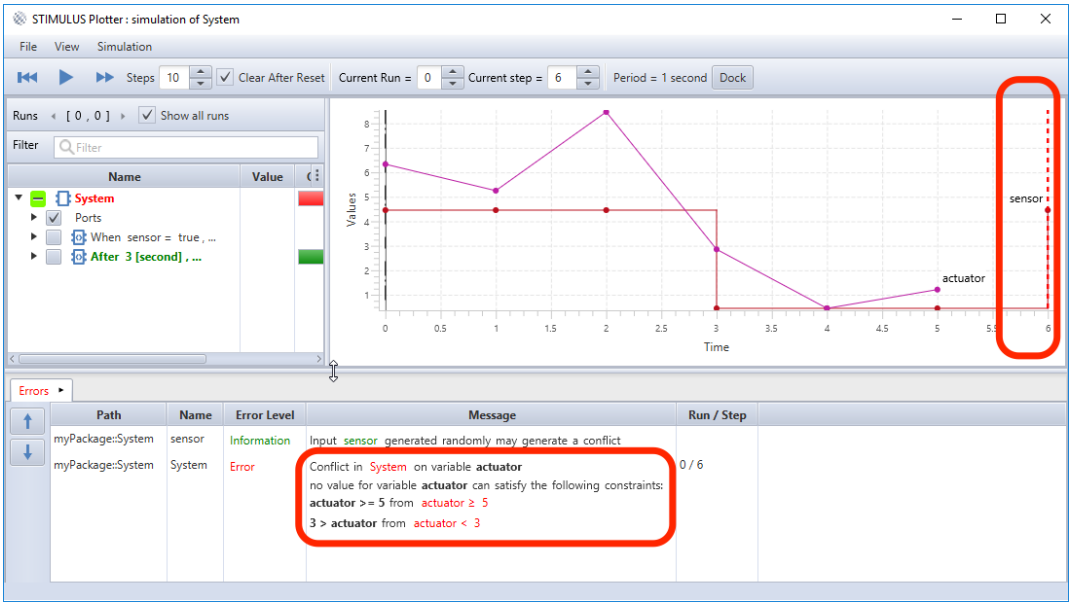

Figure 18. Simulation with a conflict

Moreover, by clicking on the links provided in the error message, you are able to get to the faulty requirements.

Question: Could you explain the error?

5. Basic notions

5.1. Predefined, customizable templates

5.2. Composition

5.3. Refinement of requirement

5.4. Observers

High level requirements can be transformed as `observers`.

6. Advanced notions

Work in progress...

6.1. Glossaries

You can host the main definitions in a `Glossary`. To start one, click on **New › glossary**

You can add the Glossary to the `Interface` (by dragging the glossary close to the little `+` close to `Glossaries`) and then add the ports to your glossary by a clicking on them (see Figure 19).

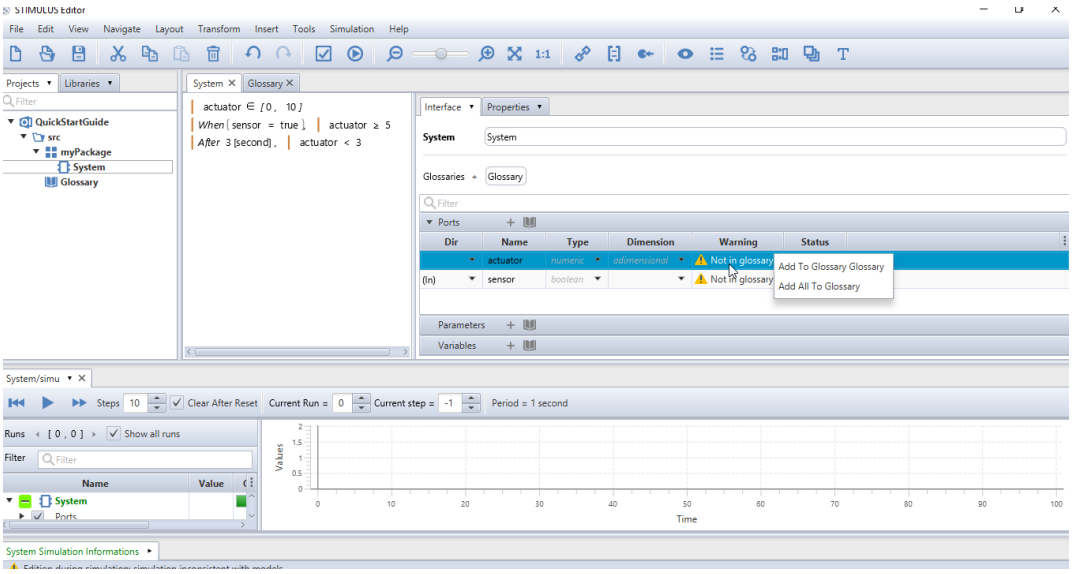

Figure 19. Adding ports to a glossary

It can be a good idea to start with the glossary

7. FAQ

7.1. Where can I find more material ?

https://www.argosim.com/free-trial/

7.2. What are the different simulation parameters ?

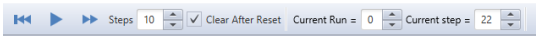

Figure 20. SimulationBar

- The **[simulate one step]** button performs one simulation step in the plot window.
- The **[fast forward button]** performs N simulation steps. The number of steps is defined by the field at the right of the button.
- The **[reset button]** starts a new simulation. When the **[Clear on reset]** checkbox is unchecked, the next simulation will be overlaid on the previous one.
- The **[Period]** label displays the simulation sampling period. You can change this setting before a simulation in the **[Properties]** panel of your system. Insert a simulation period using a dimension (e.g., *1 [second]* or *10 [millisecond]*).

7.3. How can I find a function or language construct ?

Select the `Library` folder and use the seach bar. Figure 21 illustrates the search for the `when` operator.

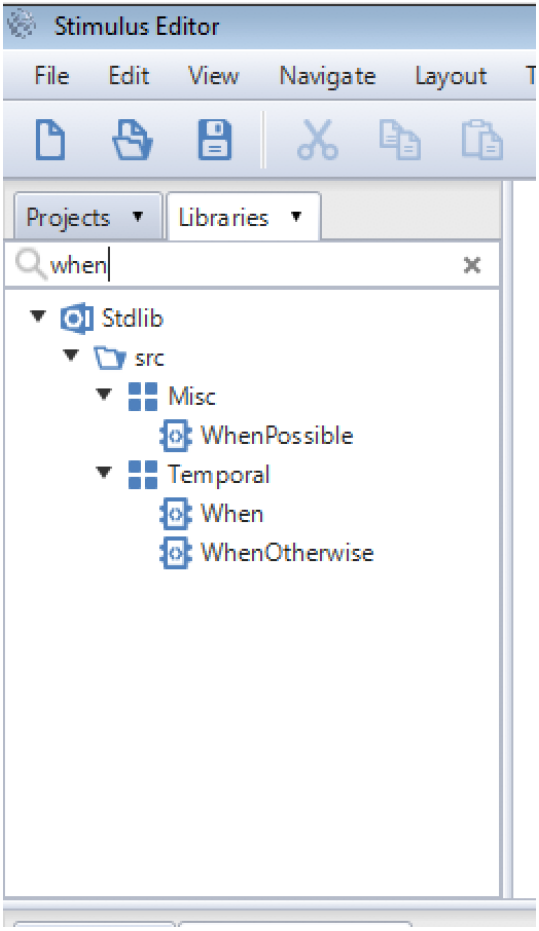

Figure 21. Search tool

7.4. How can I move a port on a system ?

7.5. How can I move a port on a system ?

To move a port in a block diagram, maintain btn:MAJ[] (Shift) while moving the port with the mouse (left click).