Acceptance testing

Felix Akkermans

11 juni 2012

1 Acceptance testing

This section describes the acceptance testing we perform after each SCRUM iteration. The acceptance tests are test if the system meets the requirements made, and how the system performs on pseudo-requirements. In particular we test these using the user interface to see if the system can perform the required tasks without problems. With these test we also manually test the user interface to see if there any display artifacts or bugs. Because these acceptance test call all the functionality throughout the whole system from the interface, they also serve as an important part of the integration testing.

For actual testing, the following table can be copied to fill in the last column with feedback. Each of the following given scenarios can be performed step by step as a test by checking if the results correspond and the actions can be performed. The tester can take liberties to deviate from the scenario and play around with the application to gain a opinion about the usability and user friendliness of the application.

Scenario	Item		Pass
1		**Basic startup and workspace initialisation:**	
	1	Start the application by going to the URL in the browser.	
	2	The browser should load the application within few seconds.	
	3	On the right of the status bar the server connection status should be resolved within a second.	
	4	The input and output gates should show by default and be able to move around by the handle bars.	
	5	The circuit filename and description should display (top right) the default string that makes it clear that they have not been specified yet. Click on the edit button in the circuit details field (top right), and the fields should be empty and only show their placeholder string. Entering values and clicking save should set the new values on display (top right).	
	6	In the status bar at the left a descriptive tooltip info string should change when hovering over different elements of the GUI.	
2		**File opening (extends scen. 1):**	
	1	Click on the file menu item, which should drop down a sub-menu with highlighting menu items when hovering over with the mouse.	
	2	In the menu click on the open item. The screen should darken and a modal dialog should slide in. Here some files should be listed.	
	3	Click open, a warning that no filename is provided should show. On clicking a file the circuit should load and the modal slide out.	
3		**File saving (extends scen. 2):**	
	4	Click the Save As item in the File menu, a modal dialog should slide in. Check that the filename of the loaded circuit is already displaying in the input field.	

Scenario	Item		Pass
	5	Click on an existing file, and a confirmation prompt should show. On a second click on the file is should save and close the dialog. Check that the filename of the circuit is updated to that of the file that was overridden. Check that the description is still the same as the old one.	
	6	When saving again, but entering a unique filename it should save immediately and close the dialog.	
	7	When saving again, but without entering a filename or entering just spaces, a warning should appear that the filename is incorrect.	
	8	Clear the circuit details if set. Click on File and then Save. Check that it should now prompts with the Save As dialog to enter a filename. Save the file and check that the provided filename now shows in the circuit details (top right).	
4		**Create simple circuit (extends scen. 1, 3):**	
	1	Drag two NOT-gates into the workspace, connect these to the inputs of an AND-gate and connect everything to the inputs and outputs.	
	2	Click on the "Choose protein" overlay on a signal. A dropdown should appear listing all proteins and a none option.	
	3	Assign random proteins to the signals. Check that the already assigned proteins are not available anymore after selecting. Check that when none is selected, the previously assigned protein is made available again for selection elsewhere.	
	4	Check that the circuit validates by clicking on the Simulate menu and selecting the Validate Circuit item. A modal dialog should appear showing in green that the circuit is valid.	
	5	Check that no two signals can be connected to a single gate input.	
	6	Check that one output signal can be assigned to different inputs.	
	7	Check that when a protein is assigned to such a forking signal, the overlay for both changes to show the same protein.	
5		**Create and simulate a 2-to-1 multiplexer (extends scen. 1):**	
	1	Drag gates into the workspace and connect them to form a 2-to-1 multiplexer. Connect the inputs and the outputs, and assign random proteins to all signals.	
	2	Open the Simulate menu and click Define Inputs. A modal dialog should appear showing graphs of each protein.	
	3	Click and drag in these graphs to define when the concentration of the proteins is high.	
	4	Define inputs so one input is selected and high at one point in time, and some later the other input is selected and high. Run the simulation and ensure that the correct output is given.	

1.1 Usability grading

Here the tester can give grades for specific and overall aspects of the usability of the application. Of course written remarks and feedback on certain aspects are also very valuable, and we encourage the tester to make these along with providing grades. On some aspects an explanation is provided in the footnotes.

In the following table, the grades correspond to the following valuations;
1 = very bad, 3 = bad, 5 = moderate/average, 7 = good, 9 = very good.

Aspect	**Score on aspect**								
Usability of circuit modelling	1	2	3	4	5	6	7	8	9
Usability of circuit management [1]	1	2	3	4	5	6	7	8	9
Responsiveness	1	2	3	4	5	6	7	8	9
Performance	1	2	3	4	5	6	7	8	9
Affordance [2]	1	2	3	4	5	6	7	8	9
Presentation of data [3]	1	2	3	4	5	6	7	8	9
Visual appeal	1	2	3	4	5	6	7	8	9
Overal application usability	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

[1] The ease of use and learability of importing, exporting, saving, opening, browsing and editing of circuit details.
[2] The degree in which the UI intuitively implies it's functionality and use.
[3] The quality of communication of, for example; simulation output data, validation results, input data, protein listings