A Flexible Tool Suite for Change-Aware Test-Driven Development of Web Applications

Esteban Robles Luna [1,3]
erobles@lifia.info.unlp.edu.ar

Juan Burella [2,3]
jburella@dc.uba.ar

Julián Grigera [1]
juliang@lifia.info.unlp.edu.ar

Gustavo Rossi [1,3]
gustavo@lifia.info.unlp.edu.ar

Process

- **1** create/update Mockups / WebSpecs
- simulate the application

derive tests

record change

- **2** Tests: affected / safe
- change object

run affected tests

- **3** Affected interaction tests

develop to make tests pass

semi-automatically update the application

- **4** develop nth set of changes / set of changes n-1 / ... / set of changes 1

verify changes

tests pass **tests fail**

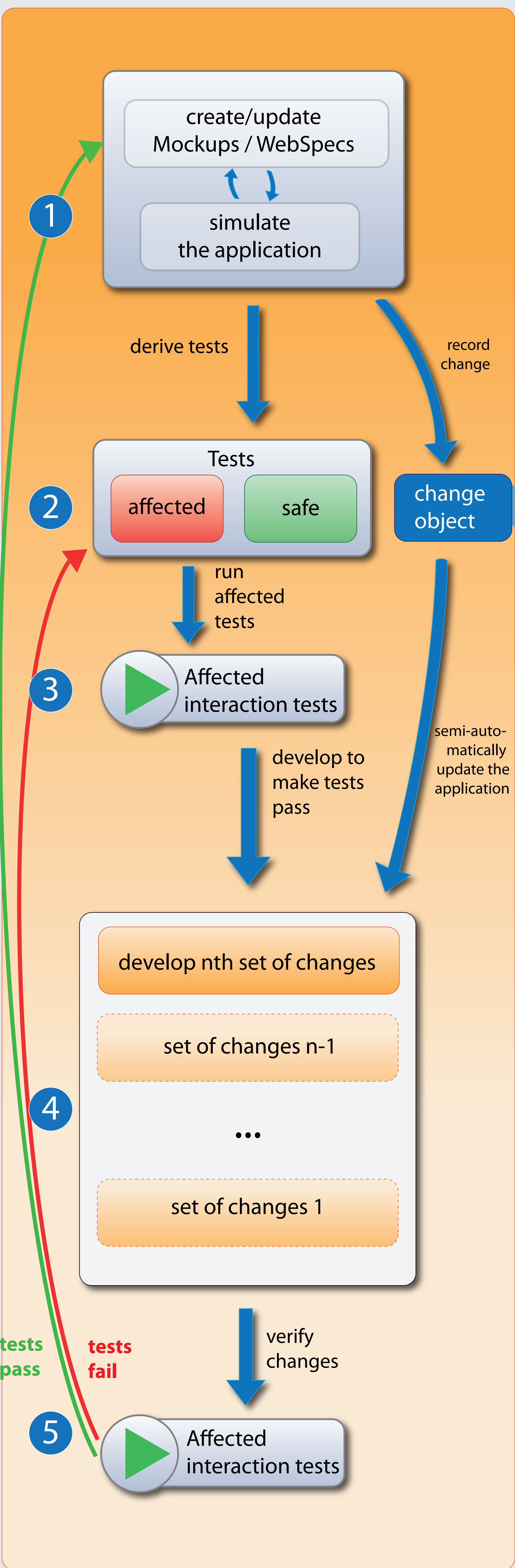- **5** Affected interaction tests

Tool Suite

Eclipse Plugin

Requirements capture

The development cycle starts by capturing requirements with Mockups and WebSpec diagrams. Mockups help to agree on the application look and feel and WebSpec allows us to specify navigation, interaction and user interface aspects in a formal and comprehensive way. WebSpec has two key elements: interactions and navigations. An interaction represents a point where the user can interact with the application by using the interaction's widgets. A diagram has a starting interaction represented with dashed lines. Some actions (like clicking a button) might produce navigation from one interaction to another. These actions are written in an intuitive DSL with the syntax: var := expr | actionName(arg1,… argn). We associate a mockup to each interaction to allow switching between the formal description and the proposed user interface while discussing with the stakeholders.
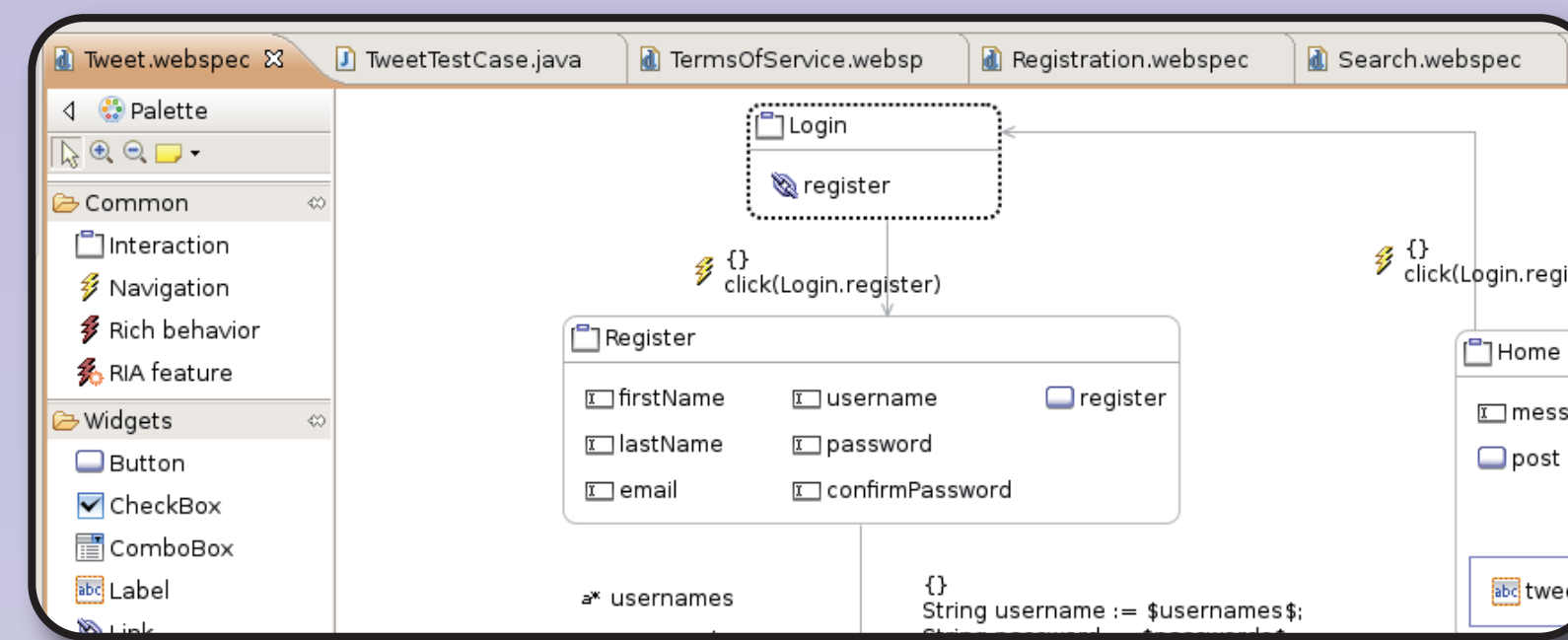
Simulation

A simulation opens a browser and reproduces a specific path executing actions (in the same way a user would do) and showing labels of the expected behavior of the application. Stakeholders can use them to propose changes before the implementation stage.
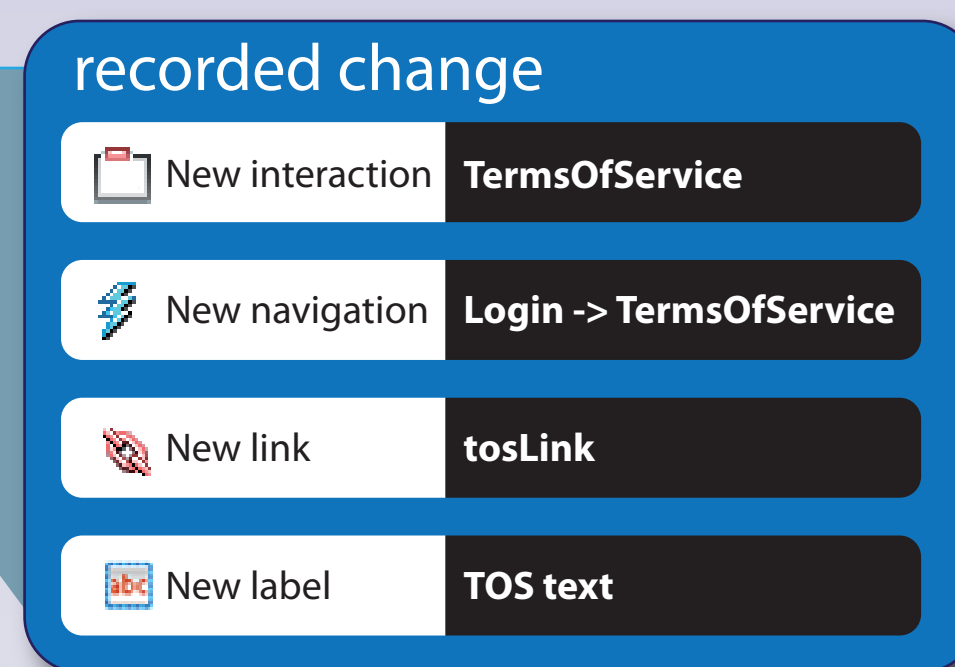
Record change

Changes made in WebSpecs are captured into change objects. WebSpecs can suffer different coarse grained changes, such as the addition or deletion of an interaction or navigation. These entities can be modified too, by the addition or deletion of widgets to an interaction, changes in invariants, etc. Regarding navigations, we can add or delete preconditions, change their source, target, or the action that triggers them.
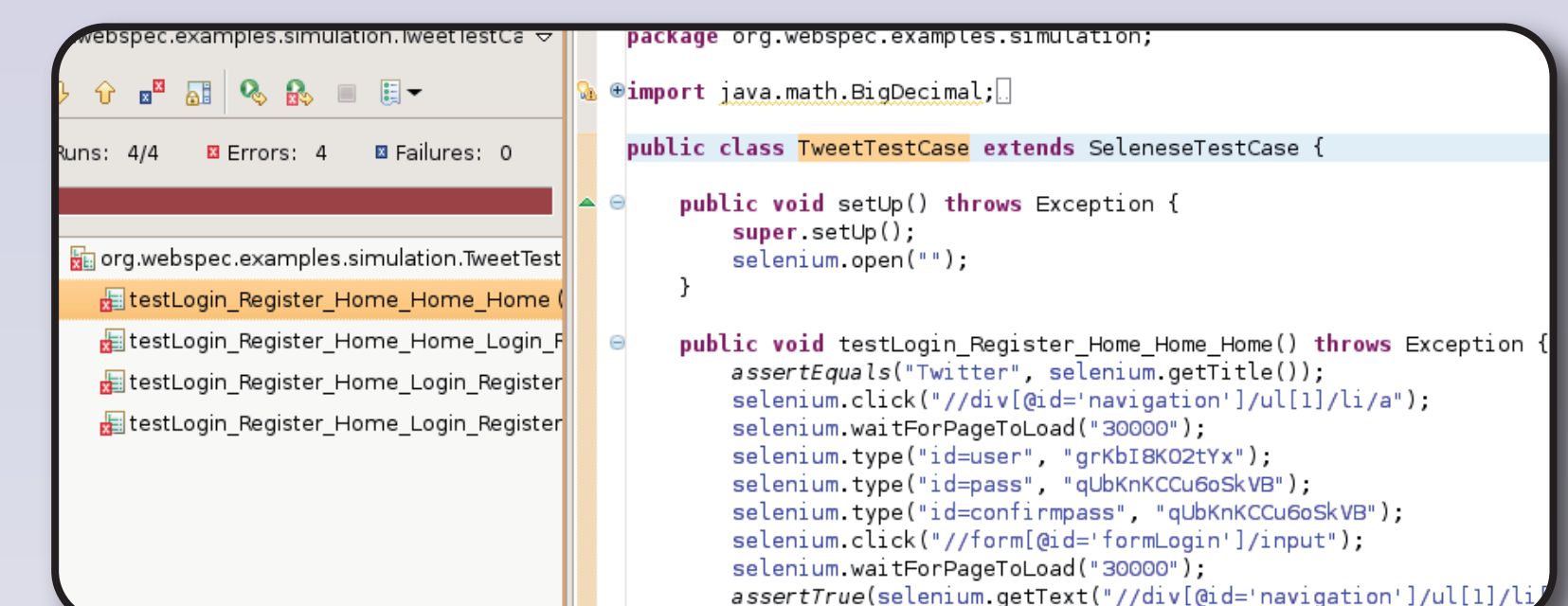
Derivated tests run

We automatically generate a set of interaction tests from the WebSpec diagram. This kind of tests allows making assertions on UI elements based on their location, so we can check the values of the different widgets. We can also automatically verify whether a requirement has been successfully implemented by validating that the application passes all tests.

Change management

Seaside effect handler

WebRatio effect handler

GWT effect handler

Development

Evolution of applications starts with changes in the requirements, and navigation/interaction requirements changes are specially frequent during the development process. All types of changes have been reified as first-class change objects that could be used to improve the tool's traceability features and automate some of these changes in the implementation. The WebSpec editor captures the changes generated and the change management tool allows the manipulation of these changes to automate their effects on concrete application's artifacts. The mechanics of these effects depend on the underlying implementation setting (GWT, WebRatio, Seaside, etc).

Eclipse Plugin

Tests run

We automatically verify whether a requirement has been successfully implemented by validating that the application passes the same tests we generated from the WebSpec diagrams to set a starting point for the development.