An Approach to Generate Test Cases from Use Cases

Javier J. Gutiérrez
University of Seville
Avd. Reina Mercedes sn.
41040 Seville, Spain
+34954553867
javierj@lsi.us.es

María J. Escalona
University of Seville
Avd. Reina Mercedes sn.
41040 Seville, Spain
+34954553867
escalona@lsi.us.es

Manuel Mejías, Jesús Torres
University of Seville
Avd. Reina Mercedes sn.
41040 Seville, Spain
+34954552769
risoto / jtorres@lsi.us.es

ABSTRACT
The system testing allows to verify the behaviour of the system under test and to guarantee the satisfaction of its requirements. This work describes a complete process to generate test cases from use cases for web applications. This process also resolves the lacks detected in existing approaches.

Categories and Subject Descriptors
D.2.5 [**Testing and Debugging**]

General Terms
Languages, Verification.

Keywords
System testing, functional testing, generation of test cases.

1. INTRODUCTION
The software systems need to improve their quality guarantee because of their growing complexity. A tool that assures the quality of software systems is the system testing. The system testing assures that the functionality of the system under test (SUT) satisfies its requirements. A system test case substitutes an actor and simulates its behaviour. This definition shows that the design of the system testing is based on the functional requirements of the system.

Nowadays, it is usual to express the functional requirements of a web system through UML use case diagrams and text templates. Existing papers [3], [6], expose that use cases and text templates are adequate for web systems. Thus, use cases are an appropriate artefact to start the generation process of test cases for web systems.

We exposed how to apply existing works in the generation of test cases in a web application in a previous paper [5]. This paper introduces a new approach to generate test cases from requirements expressed by use cases.

Actually, we know two reports that analyse and compare approaches for generating system test cases from requirements. First report [2] analyzes 12 approaches. Second report [4] analyzes 13 approaches, 4 of them also included in Denger report. A list of references to all analyzed proposals may be found in [4]. Approaches analyzed in both reports have several common characteristics: approaches work with functional requirements, functional requirements are expressed with use cases, use cases are expressed in natural language and approaches are incomplete.

The goal of the generation process is to obtain several test cases to check that all the information included in the use cases have been successfully implemented in the web system under test. The generation of a test case involves the definition of, at least, three different elements: the test values, the interactions with the system under test and the expected result. The elements obtained at the end of the generation of test cases are a set of executable test cases (with test values, interactions and expected results).

2. CASE STUDY
The system under test is a real web application that allows to manage an on-line link catalogue. We use the UML Testing Profile notation [7] to test artefacts. The use selected case is shown in table 1.

2.1 Generation of instances of use cases
First, we create a behavioural model of the use case. The UML Testing Profile does not indicate any notation for test objectives, so we use activity diagrams. Figure 2 shows this model.

Table 1. Template for the use case "Add new link".

Name	UC-01. Add new link
Main sequence	1 The user selects the option: add a new link. 2 The system selects the “top” category and shows the form to introduce the information of a link (SR-02). 3 The user introduces information of the new link and presses the insert button. 4 The system stores the new link.
Errors / alternatives	4 If the link name or URL link is empty, the system shows an error message and asks for the value again
Post condition	The new link is stored into the system

Next, we use Roundtrip pattern [1] to identify all possible paths. A path in the behavioural model is a scenario in the use case. Table in figure 1 shows some example test objectives.

2.2 Generation of test values
We use operational variables [1] to denote each concrete link that a user introduces into the system. The Category Partition method [8] is a widely used technique in existing approaches [4]. We apply this method to divide all possible links into several partitions, as showed in figure 2. We have used the Testing Profile to represent the division into partitions.

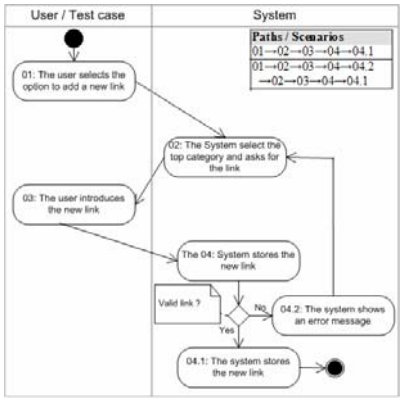

Figure 1. Behavioural model.

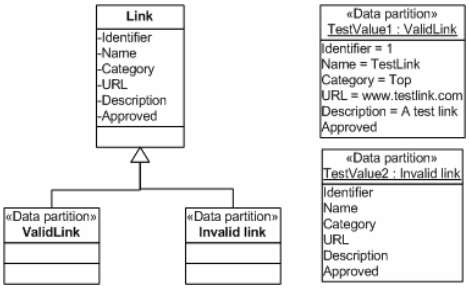

Figure 2. Test categories and test values.

Concrete test values of each category have been randomly chosen and are also shown in figure 3.

2.3 Generation of test cases

The test cases are generally described at a high level, therefore it is hard to implement test scripts. For example, the activity 02 of figure 2 could not be directly implemented. None of the existing approaches shows how to refine test objectives. We have used the sequences diagrams proposed in UML Testing Profile. The language used is an abstract language based on Canoo WebTest (webtest.canoo.com) notation. The latest is the tool used to implement our test cases.

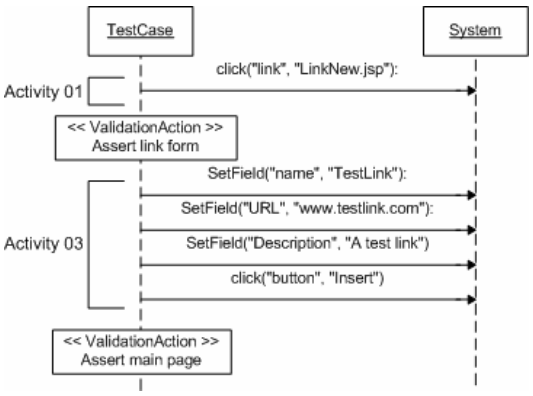

Figure 3. Test case for main scenario

Figure 3 shows the first objective (figure 1) and the validation actions that will be defined in next point.

2.4 Generation of expected results

Till now, we have developed test actions and test values. However to complete our test suite we need to define what the test case has to check and what are its expected results. The expected result for a successful operation is to reload the main page shown in figure 2. The expected result for an invalid link is an error message. The way to implement validation actions is to assert the HTML code received from the server.

3. CONCLUSIONS

This paper has presented a new approach to generate test cases from use cases. We have shown how this process is applied in a web system using use cases. All the test information has been modelled with the UML Testing profile. This profile does not describe any notation for the modelling of test objectives. We have improved it using activities diagrams derived from the use cases. It is possible and valuable for web systems to generate test cases from use cases. It is possible to start the testing process as soon as the first requirements are available or stable. It avoids postponing all the testing process till the end of the software creation, when the accumulated delays impede a deep testing. The testing design in early phases also allows to detect errors, omissions and ambiguities in the requirements when it is still easy to correct them. A prototype tool has been built. It may be downloaded from www.lsi.us.es/~javierj/

4. REFERENCES

[1] Binder, Robert V. 2000. *Testing Object-Oriented Systems*. Addison-Wesley. USA.

[2] Denger, C. Medina M. 2003. *Test Case Derived from Requirement Specifications*. Fraunhofer IESE Report.

[3] Escalona MJ. 2004. Models and Techniques for the Specification and Analysis of Navigation in Software Systems. *Ph. European Thesis*. University of Seville. Spain.

[4] Gutiérrez, J, Escalona MJ, Mejías M, Torres J. 2005. Analysis of Proposals to Generate System Test Cases From System Requirements. *CAiSE'05 Forum*. Porto. Portugal.

[5] Gutiérrez JJ, Escalona MJ, Mejías M, Torres J. 2005. A practical approach of Web System Testing. *Advances in Information Systems Development: Bridging the gap between Academia and Industry*. pp. 659-680. Ed. Springer Verlag Karlstad, Sweeden.

[6] Koch, N. *Software Engineering for Adaptative Hypermedia Applications*. Ph. Thesis, FAST Reihe Softwaretechnik Vol(12), Uni-Druck Publishing Company, Munich. 2001.

[7] Object Management Group. 2002. *The UML 2.0 Testing Profile*. www.omg.org

[8] Ostrand, TJ, Balcer, MJ. 1988. The Category-Partition Method. *Communications of the ACM*. 676-686