



# Derivation of test objectives automatically

---

Javier Gutiérrez  
María José Escalona  
Manuel Mejías  
Jesús Torres

Department of Computer Languages and Systems. University of Seville, Spain.  
[escalona@lsi.us.es](mailto:escalona@lsi.us.es)

ISD 2006. Budapest



# Introduction

## testing

### Introduction

State of the art

Use cases

A case study

Conclusions

- ➔ The growing complexity of software systems increases the need to assure their quality.
- ➔ A vital task of software development is to test the correct implementation of functional requirements.
- ➔ Some different kinds of tests can be performed: navigational testing, reliability testing, usability testing, etc.
- ➔ Our work is focussed on the functional testing.



# Introduction

## Functional testing

### Introduction

State of the art

Use cases

A case study

Conclusions

- ➔ Our test cases substitutes an actor of the system and simulates the interaction with the actor.
- ➔ The system testing requires a formal process to identify the most important test cases.

### Definition of test objectives

- ➔ This task is usually performed at the end of the software development.
- ➔ We proposed to move this task at the requirements phase in a systematic way.

Introduction

State of the art

Use cases

A case study

Conclusions

- ➔ Before starting our approach, we have identified and studied 21 approaches.
- ➔ They derived test objectives from functional requirements.
- ➔ We can classify them in several groups:
  - ✚ Depending on the artefacts used
  - ✚ Depending on the scope



# State of the art

## Depending on the artefacts

**Introduction**

**State of the art**

**Use cases**

**A case study**

**Conclusions**

Test objectives are derived from the use cases  
[Binder 1999] [Heumann 2002]

Test objectives are derived from a behavioural model  
[Nebut 2006] [Labiche 2002]

Test objectives are derived from variables and test values  
[Ostrand 1988]



# State of the art

## Depending on the scope

**Introduction**

**State of the art**

**Use cases**

**A case study**

**Conclusions**

Test objectives are derived from isolated use cases  
[Binder 1999] [Heumann 2002][Ruder 2004]

Test objectives are derived from sequences use cases  
[Nebut 2006] [Labiche 2002]

Introduction

State of the art

Use cases

A case study

Conclusions

➔ However, there is a lack in two aspects:

✚ Processes are not systematic

✚ There are not tools

➔ Our approach offers a systematic process (even automatic in some points).

**Introduction**

**State of the art**

**Use cases**

**A case study**

**Conclusions**

- ➔ The use cases is the most used technique to define functional requirements. Mainly in Web Engineering.
- ➔ OOHDM, UWE, WebML or OOH are only some example of web methodologies that proposed use cases to define web requirements.
- ➔ However, to derive system testing we need a normalization of its definition. Graphical notation is not enough.



# Use cases

## NDT to define use cases

**Introduction**

**State of the art**

**Use cases**

**A case study**

**Conclusions**

- ➔ We propose to use NDT (Navigational Development Techniques) in order to define use cases.
- ➔ NDT is a web proposal that define use cases using patterns.
- ➔ A pattern is a table with specific fields to complete them in the definition.

Introduction

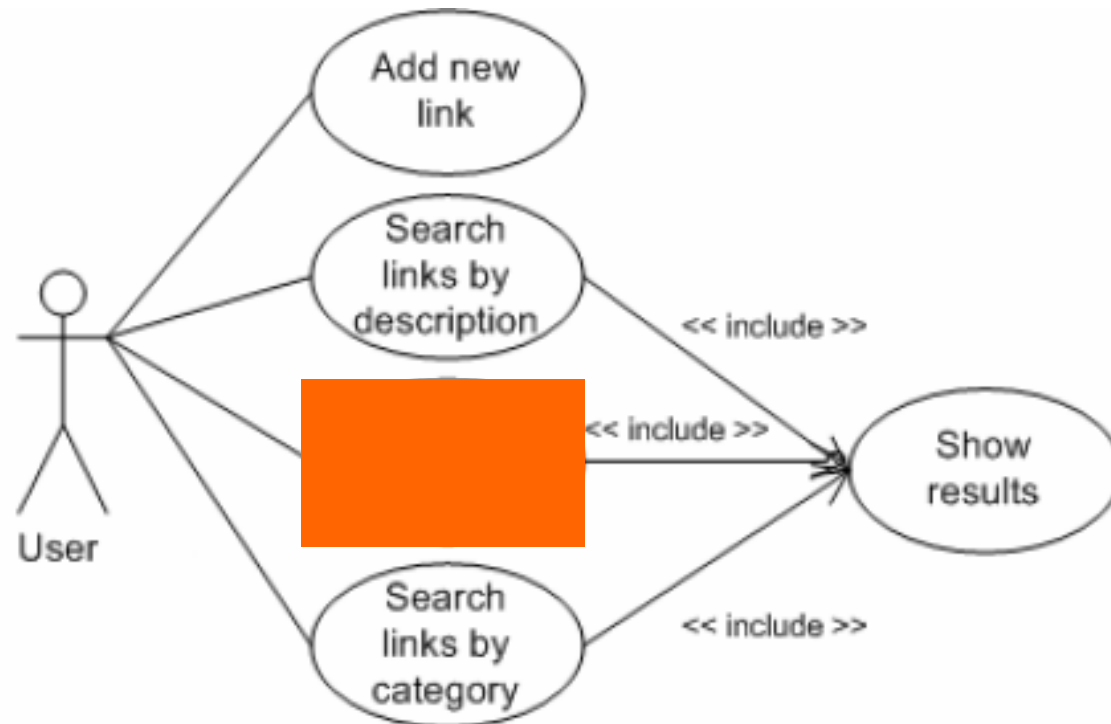
State of the art

Use cases

A case study

Conclusions

- ➔ The system under test is a web application to manage an online link catalogue ([www.codecharge.com](http://www.codecharge.com)).





# A case study

## Definition

**Introduction**

**State of the art**

**Use cases**

**A case study**

**Conclusions**

<b>Name</b>	UC-02. Search link by description
<b>Preconditions</b>	NO
<b>Main Sequence</b>	<ol style="list-style-type: none"><li>1.The user asks the system for searching links by description.</li><li>2.Te system asks for the description.</li><li>3.The user introduces the description.</li><li>4.The system searches for the links which matches up with the description introduced by the user.</li><li>5.The system shows the results.</li></ol>
<b>Errors/alternatives</b>	<ol style="list-style-type: none"><li>3.1.1. At any time, the user may cancel the search, then the use case ends.</li><li>4.1.p. If the actor introduces an empty description, then the system searches for all stored links and the result is to continue the execution of this use case.</li><li>4.2.i. If the system finds any error performing the search, then an error message is shown and this use case ends.</li><li>5.1.i. If the result is empty, then the system shows a message and this use case ends.</li></ol>
<b>Results</b>	<ol style="list-style-type: none"><li>1.The system shows the results of UC-05</li><li>3.1.i. Out of the limits of this use case.</li><li>4.2.i. Error message.</li><li>5.1.p. Message of no found results</li></ol>
<b>Post condition</b>	NO

Introduction

State of the art

Use cases

A case study

Conclusions

➔ To generate test objectives from use cases:

➔ STEP 1. Building of a behavioural model

➔ STEP 2. Deriving test objectives

➔ STEP 3. Managing the coverage of the use cases.

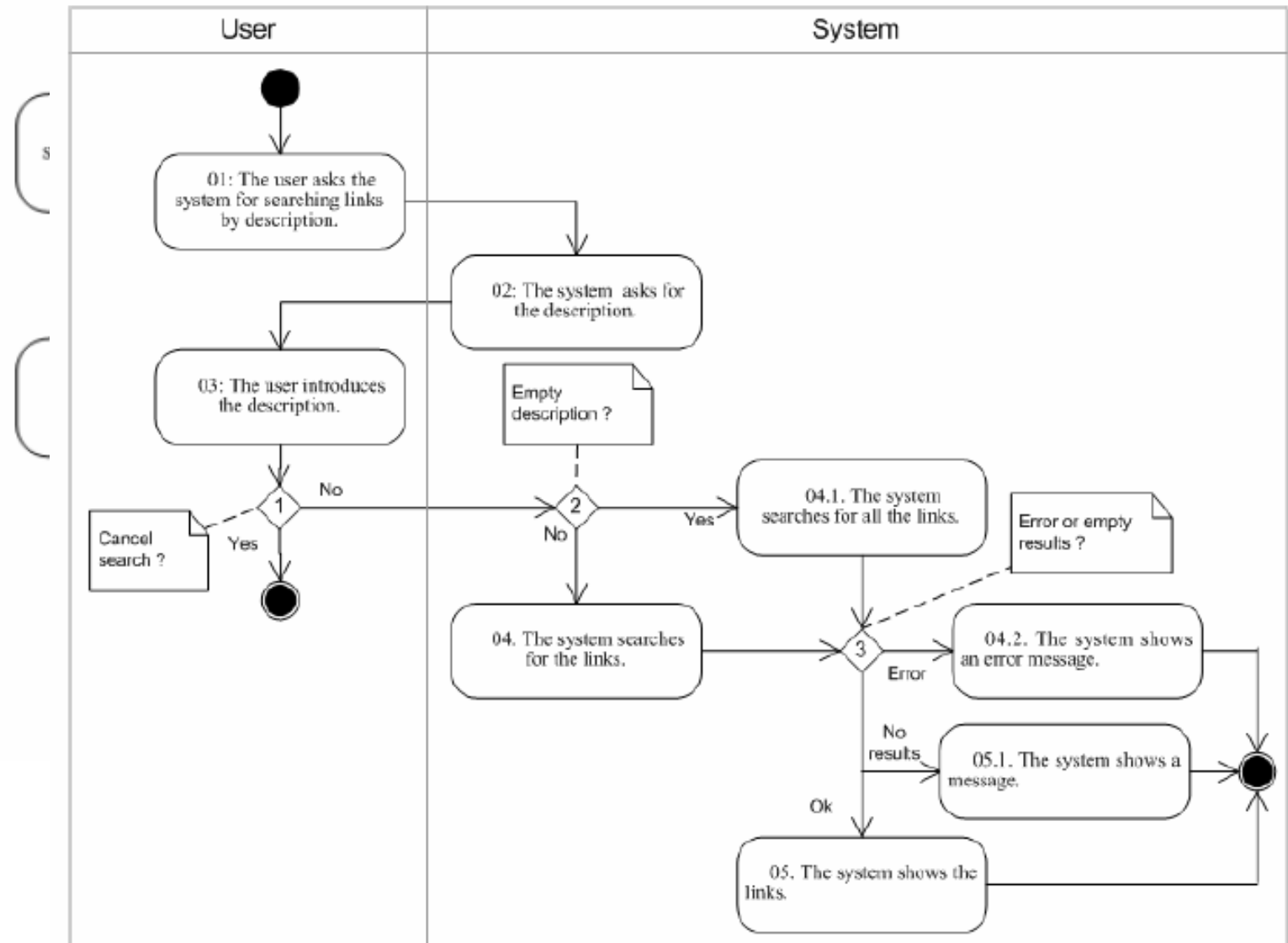
Introduction

State of the art

Use cases

A case study

Conclusions



Introduction

State of the art

Use cases

A case study

Conclusions

➔ Test objectives are systematically derived as paths over the behavioural model. It can be expressed like activity diagrams or text.

Id	Path
1	01 -> 02 -> 03 -> D1(No) -> D2(No)-> 04 -> D3(No error & Results) -> 05
2	01 -> 02 -> 03 -> D1(No) -> D2(No)-> 04 -> D3(No error & No Results) -> 05.1
3	01 -> 02 -> 03 -> D1(No) -> D2(No)-> 04 -> D3(Error) -> 04.2
4	01 -> 02 -> 03 -> D1(No) -> D2(Yes)-> 04.1 -> D3(No error & Results) -> 05
5	01 -> 02 -> 03 -> D1(No) -> D2(Yes)-> 04.1 -> D3(No error & No Results) -> 05.1
6	01 -> 02 -> 03 -> D1(No) -> D2(Yes)-> 04.1 -> D3(Error) -> 04.2
7	01 -> 02 -> 03 -> D1(Yes)



# A case study

## Derivation of test objectives

Introduction

State of the art

Use cases

A case study

Conclusions

- ➔ However, it must be enriched with test values and expected results.
- ➔ After that, using a tool like an implementation of the Algorithm **BuildTestObjectives**.

Introduction

State of the art

Use cases

A case study

Conclusions

- ➔ The coverage measures the number of scenarios from a use case with an attached test objective.
- ➔ A higher coverage implies more test objectives and more test cases.
- ➔ It can be calculated like:

$$\frac{\text{Number of test objectives}}{\text{Number of scenarios}} = \text{Test coverage}$$

The coverage might be determined by the relevance of frequency of the use case.

Introduction

State of the art

Use cases

A case study

Conclusions

- ➔ Our approach offers a systematical way to generate test cases from the requirements model.
- ➔ It gives models and algorithm that can be implemented.
- ➔ A prototype of a tool is available [www.lsi.us.es/javierj](http://www.lsi.us.es/javierj)



# Derivation of test objectives automatically

---

**Thank you very much!!!**

Javier Gutiérrez  
María José Escalona  
Manuel Mejías  
Jesús Torres