Traceability From Use Cases To Test Cases

Peter Zielczynski, Ph.D.
The “A” Consulting Team
PZielczynski@tact.com

Agenda

- Overview of the requirements types
- Traceability between requirements
- Actors and Use Cases
- How to create Test Cases from Use Cases
- Mapping to RUP
- Conclusions
What is a requirement?

- A requirement is defined as "a condition or capability to which a system must conform"
  - A capability needed by a customer or user to solve a problem or achieve an objective
  - A capability that must be met or possessed by a system to satisfy a contract, standard, specification, regulation or other formally imposed document
  - A restriction imposed by a stakeholder

**Requirements pyramid**

- Needs
- Features
- Use Cases
- Scenarios
- Test Cases
Agenda

- Overview of the requirements types
- Traceability between requirements
- Actors and Use Cases
- How to create Test Cases from Use Cases
- Mapping to RUP
- Conclusions

What is traceability?

- A technique used to provide relationships between requirements, design and implementation of a system in order to manage the effect of change and ensure the success of the delivered systems
- The ability to describe and follow the life of a requirement, in both a forward and backward direction
- A technique allowing to determine the origins of any requirement
**Traceability of requirements**

- Needs
- Features
- Use Cases / Supp Spec
- Scenarios
- Test Cases

**Role of traceability**

- To verify that implementation fulfills requirements
  - Lower-level requirements are consistent with stakeholder needs
- To verify that the application does only what was intended
  - All functionality was requested
  - Only required testing was performed
- To help change management
  - Analyze impact of requirements change
What is a traceability item?

- Any project element that needs to be explicitly traced from another textual or model item in order to keep track of the dependencies between them.

- With respect to RequisitePro: Any project element represented within RequisitePro by an instance of a RequisitePro requirement type.

- Sample traceability items defined as requirement types in RequisitePro:
  - Stakeholder Need
  - Feature
  - Use Case
  - Actor
  - Glossary Term

Showing traceability in RequisitePro
Agenda

- Overview of the Requirements Types
- Traceability between requirements
- Actors and Use Cases
- How to create Test Cases from Use Cases
- Mapping to RUP
- Conclusions

Use Cases and Actors

- **Use Case**: A description of system behavior, in terms of sequences of actions. A use case should yield an observable result of value to an actor. A use case contains basic flow and all related alternate flows of events.

- **Actor**: Someone or something external that interacts with the system.
Characteristics of a Use Case

- Is initiated by an actor to invoke functionality
- Models an interaction between an actor and the system
- Describes sequence of actions or related steps
- Captures functional requirements
- Provides value to the actor
- Is a complete and meaningful flow of events from the perspective of a particular actor

ABC of writing effective Use Cases

- Address functionality
- Briefness in narrative
- Communicate effectively

(By Anand Maharana)
What is a purpose of Use Cases?

- To agree with the customers and users on what the system should do
- Basis for Use Case realizations that play major role in the design
- May be almost directly used for documentation
- Basis for planning the technical content of iterations
- Give system developers a better understanding of the purpose of the system
- Can be used as an input for Test Cases

Use Case Diagrams

[Diagram showing use cases for placing and canceling orders, and checking order status with a customer as the actor.]
Use Case format

1. Brief Description
2. Flow of Events
   1. Basic Flow
   2. Alternative Flow 1
   3. Alternative Flow 2
3. Special Requirements
4. PreConditions
5. PostConditions
6. Extension Points
7. Context Diagram
8. Activity Diagram

Use Case “Place an order”: Basic Flow

B1 User enters web site address in the browser.
System displays login page.

B2 User enters an email address and a password.
System confirms correct login, presents main page, and prompts for a search string.

B3 User enters search string – partial name of a book.
System returns all books matching search criteria.

B4 User selects a book.
System presents detailed information about a book.

B5 User adds the book to a shopping cart.
Shopping cart contents is presented to the user.

B6 User selects “proceed to checkout” option.
System asks for confirmation of a shipping address.
Use Case “Place an order”: Basic Flow

B7  User confirms shipping address.
    System presents shipping options.
B8  User selects shipping option.
    Systems asks which credit card will be used.
B9  User confirms credit card that is stored in the system.
    System asks for final confirmation to place an order.
B10 User places the order.
    System returns confirmation number.

Alternate Flow 1: “Unregistered user”

A1.1 Alternate flow begins after step B1 when user selects “Register now” option.
    System prompts for required information: name, address, userid, password.
A1.2 User provides required information.
    Systems confirms registration and presents login screen.
Alternate Flow 2: “Invalid password”

A2.1 Alternate flow begins after step B1 when user enters userid and password. When either userid or password are invalid, system returns an error message and displays login page.

A2.2 Flow continues from step B2 of the Basic Flow.

Activity Diagram

A1 – unregistered user

A2 – invalid password

A3 – no books found

A4 – declining a book

A5 – continue shopping
Activity Diagram

A6 – new address
- Is shipping address the same as in previous order?
  - Yes
    - User confirms address
    - User selects shipping options
  - No
    - User enters new address

A7 – new credit card
- Does user want to use same credit card?
  - Yes
    - User confirms the card
  - No
    - User adds new credit card information

A8 – cancel order
- Does user want to cancel the order?
  - Yes
    - User cancels order
  - No
    - User confirms order
    - System returns confirmation number

Agenda

- Overview of the requirements types
- Traceability between requirements
- Actors and Use Cases
- How to create Test Cases from Use Cases
- Mapping to RUP
- Conclusions
Proposed format prepares generation of Test Cases

- Steps in a Use Case correspond to steps in Test Case
- Unique numbering makes referring steps easy
- Combining user’s input and system’s response in one step
- Activity diagram is a basis for scenario generation

What is a Scenario?

- Scenario describes one particular path through the flow of events described in a Use Case
- Scenario is an instance of an Use Case
How to find all Scenarios

Description of Scenario

Scenario can be described by:

- A sequence of alternate flows
  - Example:
    SC16: A2, A2, A6

- A sequence of steps that form a scenario
  - Example:
    SC16: B1, A2.1, A2.2, A2.1, A2.2, B2, B3, B4, B5, B6, A6.1, A6.2, B8, B9, B10
Infinite loops

Scenarios for book ordering use case

SC1 – B
SC2 – A1
SC3 – A2
SC4 – A3
SC5 – A4
SC6 – A5
SC7 – A6
SC8 – A7
SC9 – A8
SC10 – A1, A2
SC11 – A3, A4
SC12 – A4, A5
SC13 – A3, A5
SC14 – A6, A7
SC15 – A7, A8

2^{(8+4)} = 4096

A1 – unregistered user
A2 – invalid password
A3 – no books found
A4 – declining a book
A5 – continue shopping
A6 – new address
A7 – new credit card
A8 – cancel order
Create requirement type “Scenario”

Traceability from Use Cases to Scenarios
From Scenarios to Test Cases

- Identify variables for each Use Case step
- Identify significantly different options for each step
- Select options that should be tested
- Combine options to be tested into Test Cases
- Assign values to variables

Identify variables for each Use Case step

- B2  Enter e-mail address and password  
  - email (string)
  - password (string)
- B3  Search a book  
  - search sentence (string)
- B4  Select a book  
  - books (list)
- B8  Select shipping option  
  - shipping method
Identify significantly different options for each step

- **Strings**
  - regular: john_smith@aol.com
  - empty
  - min allowed: a@b.com
  - max allowed: 12345678910@verylongdomainname.com
  - max + 1: 012345678910@verylongdomainname.com
  - very big: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa...
  - logically invalid: john_smith@aol.xyz

- **Characters**
  - lowercase: a
  - uppercase: A
  - numeric: 5
  - special: #

- **Numbers**
  - regular
  - min
  - max
  - min - 1
  - max + 1
  - 0
  - Negative

- **Search criteria**
  - 1 element returned
  - many elements
  - no elements
Identify significantly different options for each step

- Single selection sets of items
  - each item selected once

- Multiple selection sets of items
  - one selection
  - no selection
  - many selections
  - all items selected

Select options that should be tested

- B1 email: regular, empty, 1, 50, 51, 257 char, no @
- B1 password: regular, empty, 5, 6, 10, 11, 257 char
- B2 search sentence: regular, empty, 1, 300, 301 char
- B3 books: first selection, last selection
- B6 shipping method: each method once
Combine options to be tested into Test Cases

<table>
<thead>
<tr>
<th>email</th>
<th>pwd</th>
<th>search</th>
<th>book</th>
<th>add</th>
<th>checkout</th>
<th>shipping</th>
<th>order</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>10</td>
<td>300</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>11</td>
<td>301</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Test Case 1
Test Case 2
Test Case 3
Test Case 4
Guidelines for combining options

Rules to select an option is step n:

- Take any option that was not used yet
- Take an option that was not used after the option selected in step n-1
- Take an option that was used the least

Test Case matrix

<table>
<thead>
<tr>
<th>Step</th>
<th>Variable or selection</th>
<th>TC1</th>
<th>TC2</th>
<th>TC3</th>
<th>TC4</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1</td>
<td>Website</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
</tr>
<tr>
<td>B2</td>
<td>Email</td>
<td>Regular</td>
<td>Min</td>
<td>Max</td>
<td>Regular</td>
</tr>
<tr>
<td>B2</td>
<td>Password</td>
<td>Regular</td>
<td>Min</td>
<td>Max</td>
<td>Min</td>
</tr>
<tr>
<td>B3</td>
<td>Search string</td>
<td>Regular</td>
<td>Min</td>
<td>Max</td>
<td>Regular</td>
</tr>
<tr>
<td>B4</td>
<td>Book selection</td>
<td>First</td>
<td>Last</td>
<td>First</td>
<td>Last</td>
</tr>
<tr>
<td>B5</td>
<td>Shopping chart</td>
<td>Add</td>
<td>Add</td>
<td>Add</td>
<td>Add</td>
</tr>
<tr>
<td>B6</td>
<td>Checkout</td>
<td>Checkout</td>
<td>Checkout</td>
<td>Checkout</td>
<td>Checkout</td>
</tr>
<tr>
<td>B7</td>
<td>Address</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td>B8</td>
<td>Shipping option</td>
<td>1\textsuperscript{st}</td>
<td>2\textsuperscript{nd}</td>
<td>3\textsuperscript{rd}</td>
<td>4\textsuperscript{th}</td>
</tr>
<tr>
<td>B9</td>
<td>Credit card</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
</tbody>
</table>
Assign values to variables

<table>
<thead>
<tr>
<th>Step number</th>
<th>Variable or selection</th>
<th>Value</th>
<th>Expected result</th>
<th>Obtained result</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1</td>
<td>Website</td>
<td><a href="http://www.amazon.com">www.amazon.com</a></td>
<td>Logon Screen</td>
<td></td>
</tr>
<tr>
<td>B2</td>
<td>Email</td>
<td><a href="mailto:Jsmith@hotmail.com">Jsmith@hotmail.com</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B3</td>
<td>Password</td>
<td>Johnsm</td>
<td>Main Screen</td>
<td></td>
</tr>
<tr>
<td>B4</td>
<td>Search string</td>
<td>“Rational”</td>
<td>List of books</td>
<td></td>
</tr>
<tr>
<td>B5</td>
<td>Book selection</td>
<td>First</td>
<td>Book details</td>
<td></td>
</tr>
<tr>
<td>B6</td>
<td>Shopping chart</td>
<td>Shopping cart</td>
<td>Cart contents</td>
<td></td>
</tr>
<tr>
<td>B7</td>
<td>Checkout</td>
<td>Checkout</td>
<td>Prompt for address</td>
<td></td>
</tr>
<tr>
<td>B8</td>
<td>Address</td>
<td>Same</td>
<td>Prompt for shipping</td>
<td></td>
</tr>
<tr>
<td>B9</td>
<td>Shipping option</td>
<td>5 days</td>
<td>Prompt for payment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Credit card</td>
<td>Same</td>
<td>Prompt for confirmation</td>
<td></td>
</tr>
</tbody>
</table>
Traceability tree
Suspect relationships

What is being tested
Agenda

- Overview of the requirements types
- Traceability between requirements
- Actors and Use Cases
- How to create Test Cases from Use Cases
- Mapping to RUP
- Conclusions
Where in the process does it fit?

**Agenda**

- Overview of the requirements types
- Traceability between requirements
- Actors and Use Cases
- How to create Test Cases from Use Cases
- Mapping to RUP
- Conclusions
Benefits of this approach

- Test Cases are derived in more automatic way
- Avoiding duplicate testing
- Better test coverage
- Easier monitoring of testing progress
- Easier work load balancing between testers
- Easier regression testing
- Decreasing project time by moving some tasks from construction to elaboration
- Contribution to early discovery of missing requirements

Literature

- Jim Heumann, "From Use Cases to Test Cases - Ensuring Quality from the Beginning." RUC 2001.
QUESTIONS

Thank You

Peter Zielczynski, Ph.D.
PZielczynski@tact.com