IVAR JACOBSON INTERNATIONAL

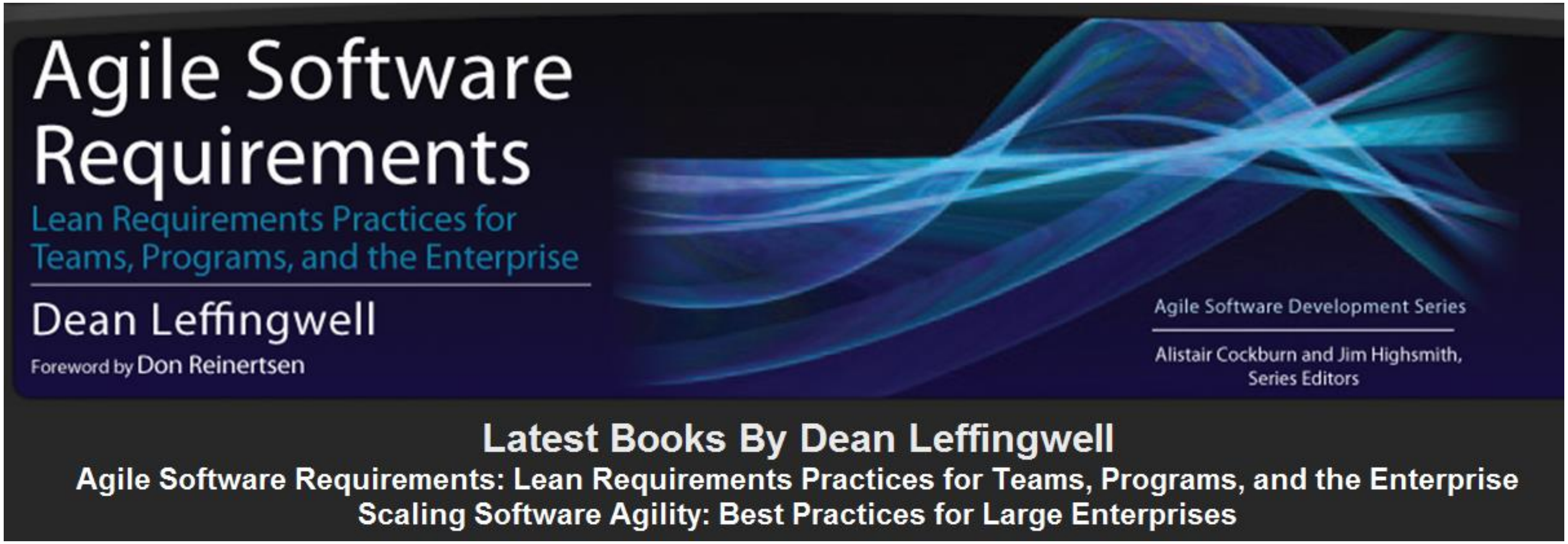
THE SMARTER WAY

Use-Case Model

Note: Ivar Jacobson International course material is for classroom use only

www.ivarjacobson.com

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved



Agile Software Requirements

Lean Requirements Practices for Teams, Programs, and the Enterprise

Dean Leffingwell

Foreword by Don Reinertsen

Agile Software Development Series

Alistair Cockburn and Jim Highsmith, Series Editors

Latest Books By Dean Leffingwell

Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise

Scaling Software Agility: Best Practices for Large Enterprises

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

User Story Template

- I <in the role of XX> needs functionality <zzz> to achieve the goal of <YYY>

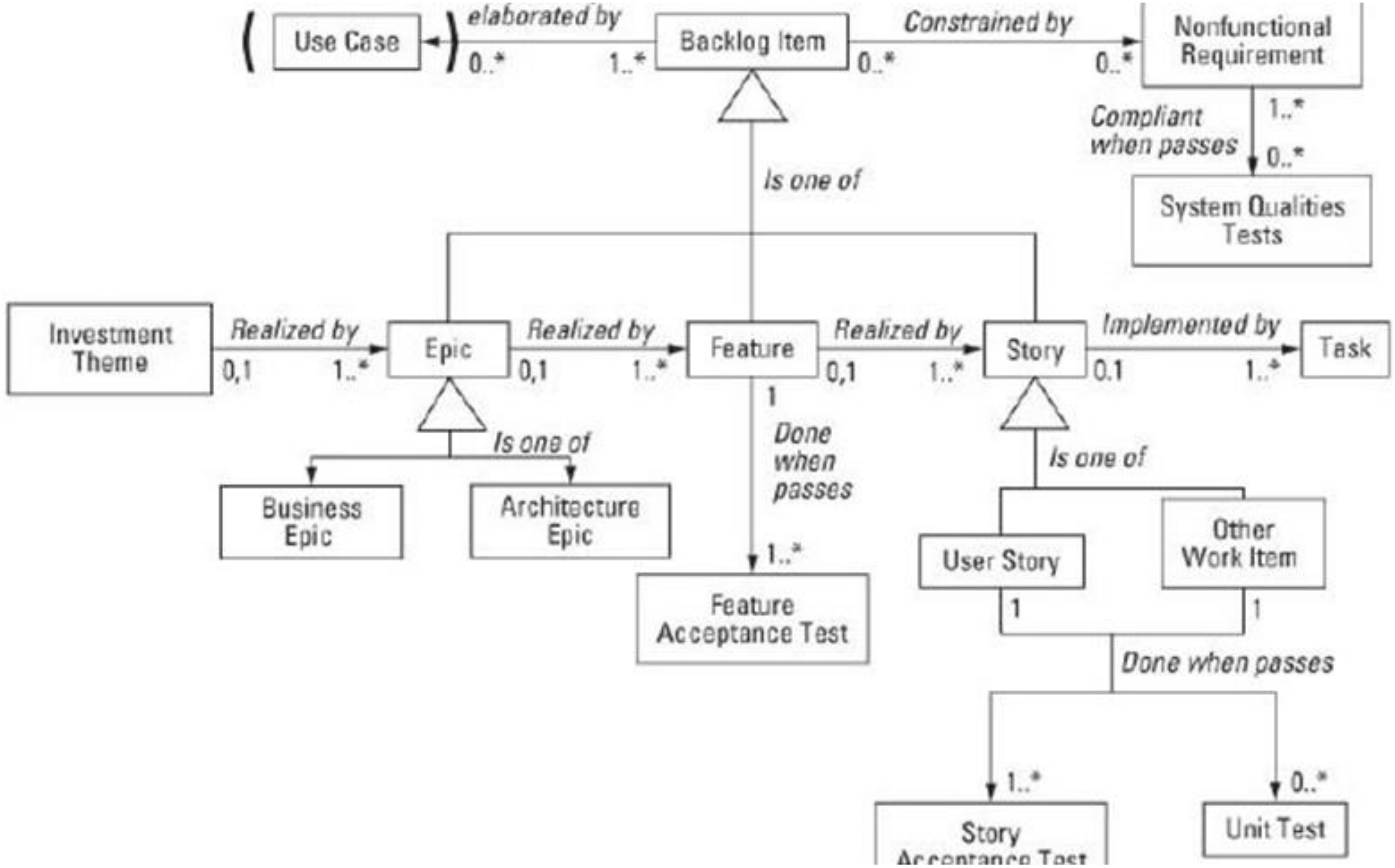
- User storys first
  – then further enhancements with a use case template

- Recommendations:

- Start with a list of user stories
- Enhance the most important/complex user stories (with priorities in the backlog) with use case model
- See next metamodel

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Backlog metamodel



IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

THE SMARTER WAY

Use-Case 2.0

Module 2 – Finding Actors and Use Cases

www.ivarjacobson.com

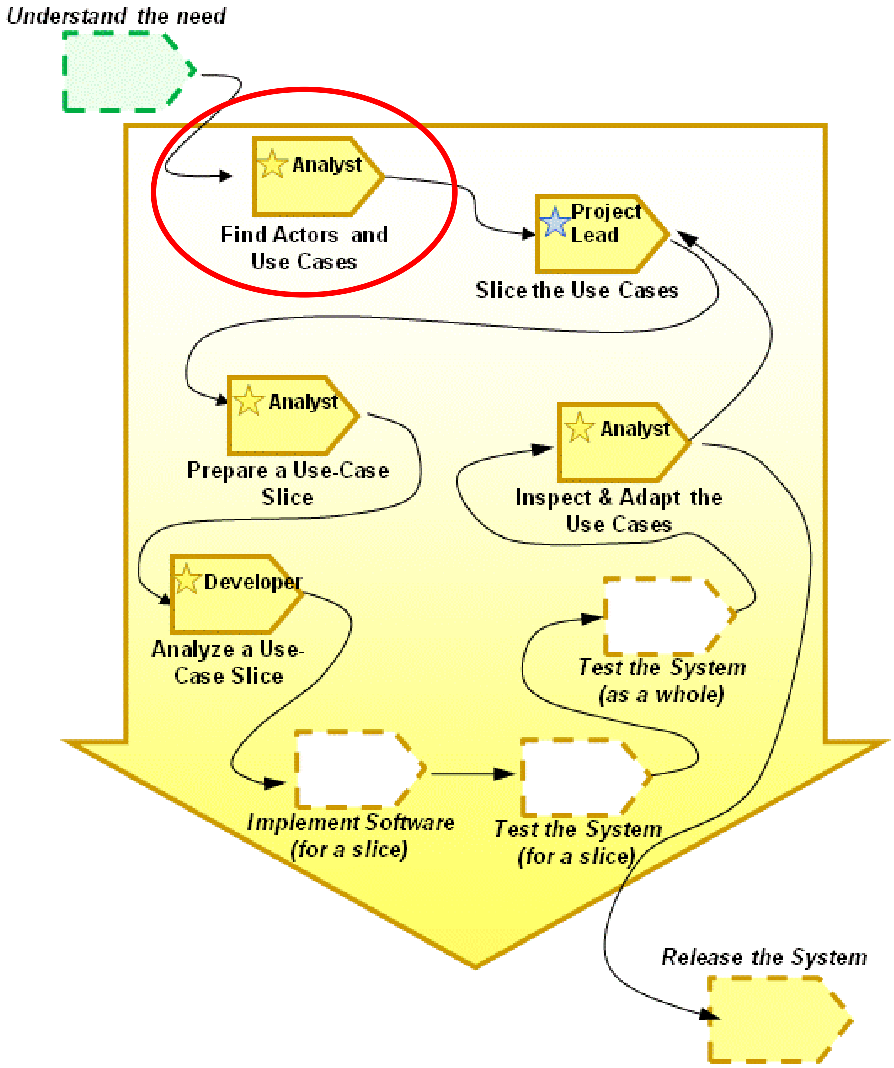
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

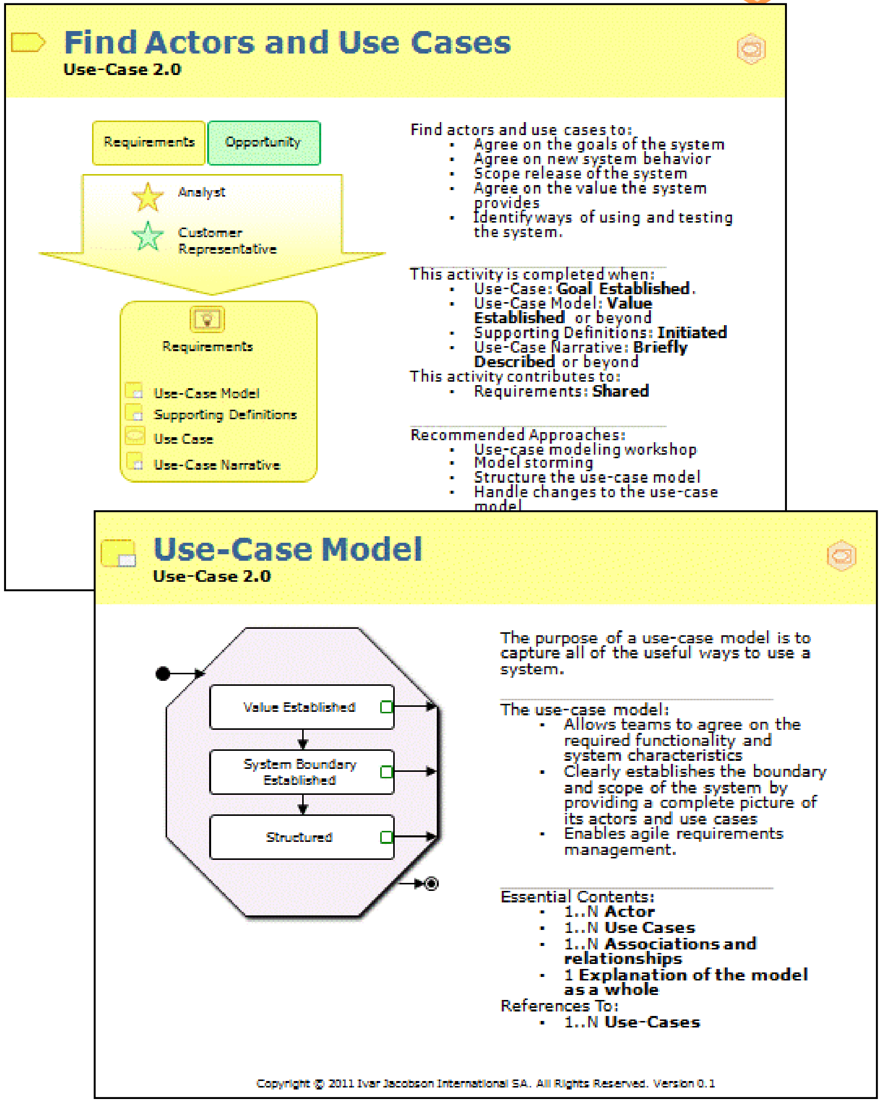
Objectives

- Understand the system boundary
- How to identify primary and supporting actors
- How to name and describe actors
- Identifying primary and supporting use cases
- How to name and describe use cases

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Where are we?





Find Actors and Use Cases

Use-Case 2.0

Find actors and use cases to:
- Agree on the goals of the system
- Agree on new system behavior
- Scope release of the system
- Agree on the value the system provides
- Identify ways of using and testing the system.

This activity is completed when:
- Use-Case: **Goal Established**.
- Use-Case Model: **Value Established** or beyond
- Supporting Definitions: **Initiated**
- Use-Case Narrative: **Briefly Described** or beyond

This activity contributes to:
- Requirements: **Shared**

Recommended Approaches:
- Use-case modeling workshop
- Model storming
- Structure the use-case model
- Handle changes to the use-case model

Use-Case Model

Use-Case 2.0

The purpose of a use-case model is to capture all of the useful ways to use a system.

The use-case model:
- Allows teams to agree on the required functionality and system characteristics
- Clearly establishes the boundary and scope of the system by providing a complete picture of its actors and use cases
- Enables agile requirements management.

Essential Contents:
- 1..N **Actor**
- 1..N **Use Cases**
- 1..N **Associations and relationships**
- 1 **Explanation of the model as a whole**

References To:
- 1..N **Use-Cases**

Copyright © 2011 Ivar Jacobson International SA. All Rights Reserved. Version 0.1

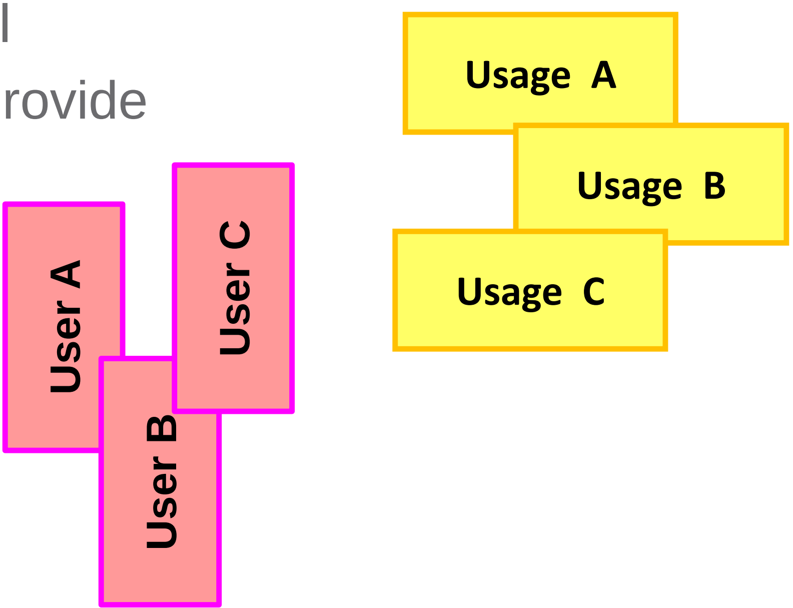
IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Exercise 2.1: Getting Started by Model Storming

- In your group:
  - Identify a target system
  - Brainstorm as many candidate *users* as possible
    - Anything that interacts with the system
    - This could be people or it could be other systems
  - Brainstorm as many candidate system *usages* as possible
    - Any goal the system can fulfill
    - Any service the system can provide



User A

User B

User C

Usage A

Usage B

Usage C

IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

From model storm to use-case model

Use-Case Model contains *Actors* and *Use Cases*

Actor

- Someone or something outside the system that interacts with the system in a particular role
- A representation of one or more stakeholders who use the system and contribute to the completion of their goals

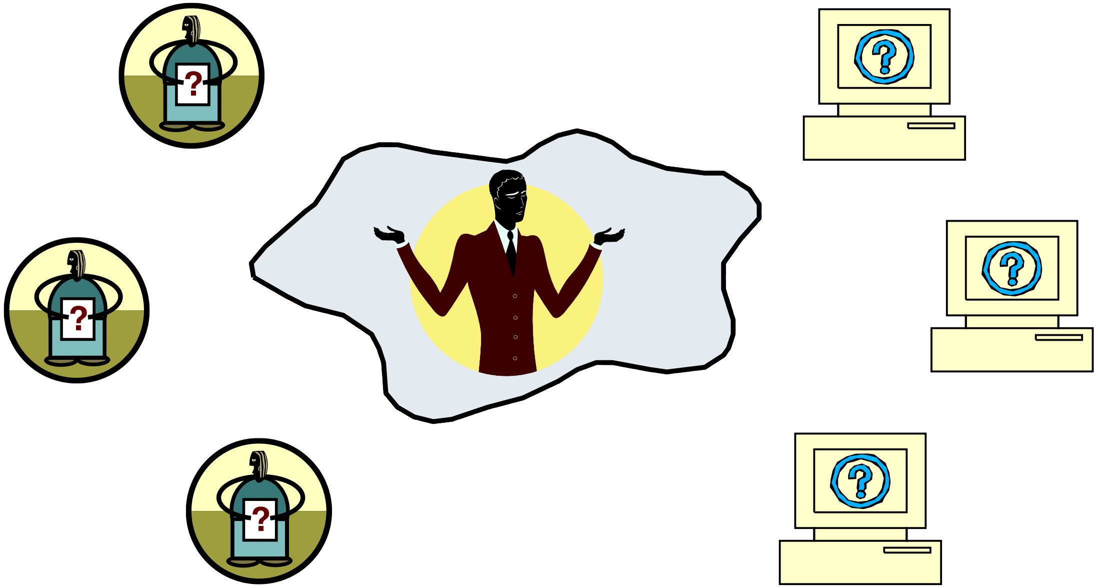
Use Case

- All the ways of using a system to achieve a particular goal for a particular user.

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

First scope the system

- Understand your system boundary:
  - Everything beyond this that interacts with the system is an actor
  - Use a context diagram to help the thought process

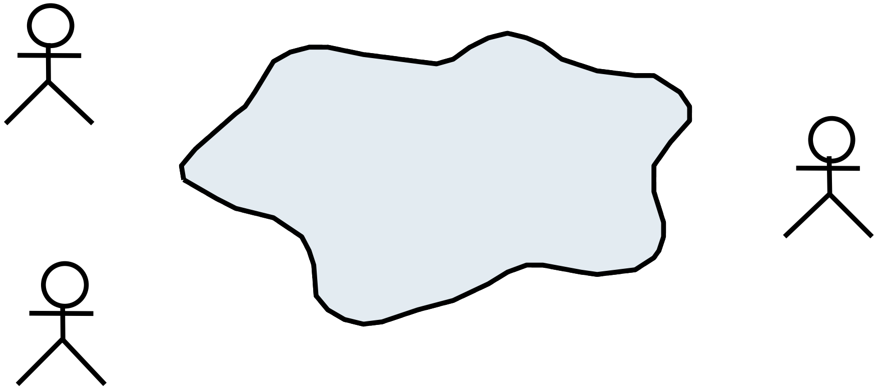


Start by identifying the users and connected systems.

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Find the most important actors

- Identify the actors that will get the most value from the system
  - These are the actors for which the system is built



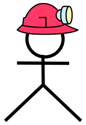
Find the primary actors.

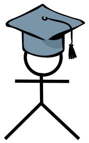
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

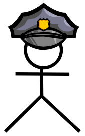
Leverage the list of users and other systems

- For each user or system identified:
  - Ensure that there is at least one Actor defined to address their needs of the system
- The actors that represent the roles adopted by the key users are the primary actors

User Types	Actors
Technology Adopter	Calling Subscriber, Callable Subscriber, Customer
Standard User	Calling Subscriber, Callable Subscriber, Customer
Messaging Devices	Calling Subscriber, Callable Subscriber


Calling Subscriber


Callable Subscriber

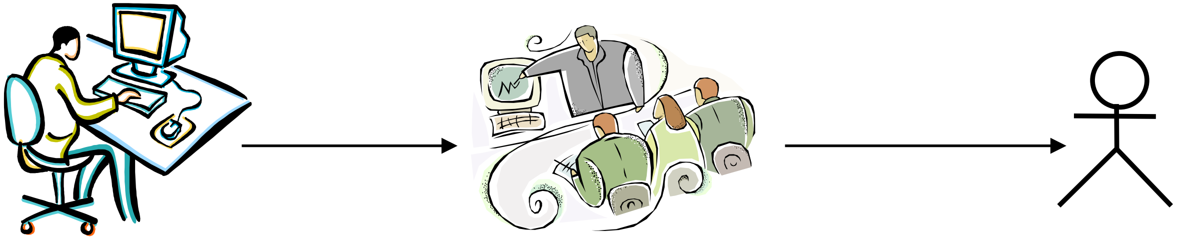

Customer

Remember: ***Actors are roles played with respect to the system***

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Work from the specific to the general

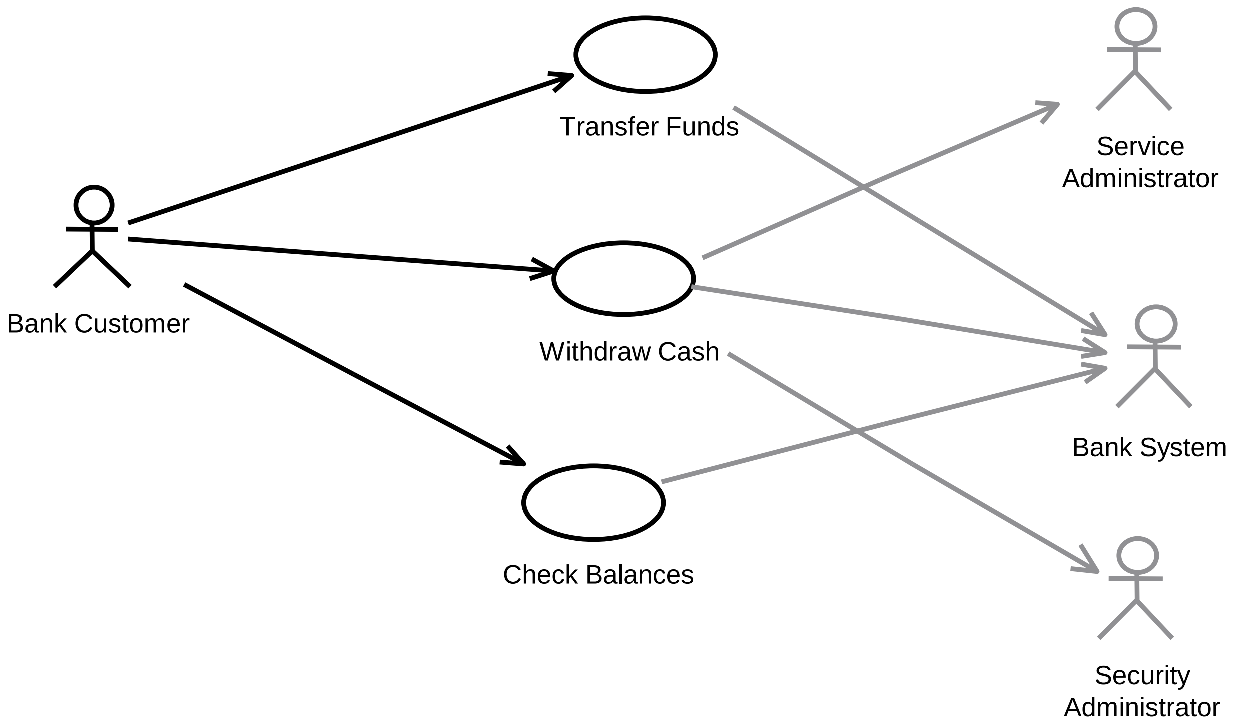
- Don’t try and pluck the most suitable actor name ‘out of the air’ in a brainstorming session:
  - Work down to a good actor name by first identifying a few people who play the same role with respect to the system
  - Don’t get pre-occupied with the ‘perfect actor name’, ‘good enough’ is sufficient
- If you get stuck think of real individuals who will use the system to help the thought process



Work from individuals to user types to actors if you need to

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Include the secondary and supporting actors



Does the ATM need any help when supplying cash?

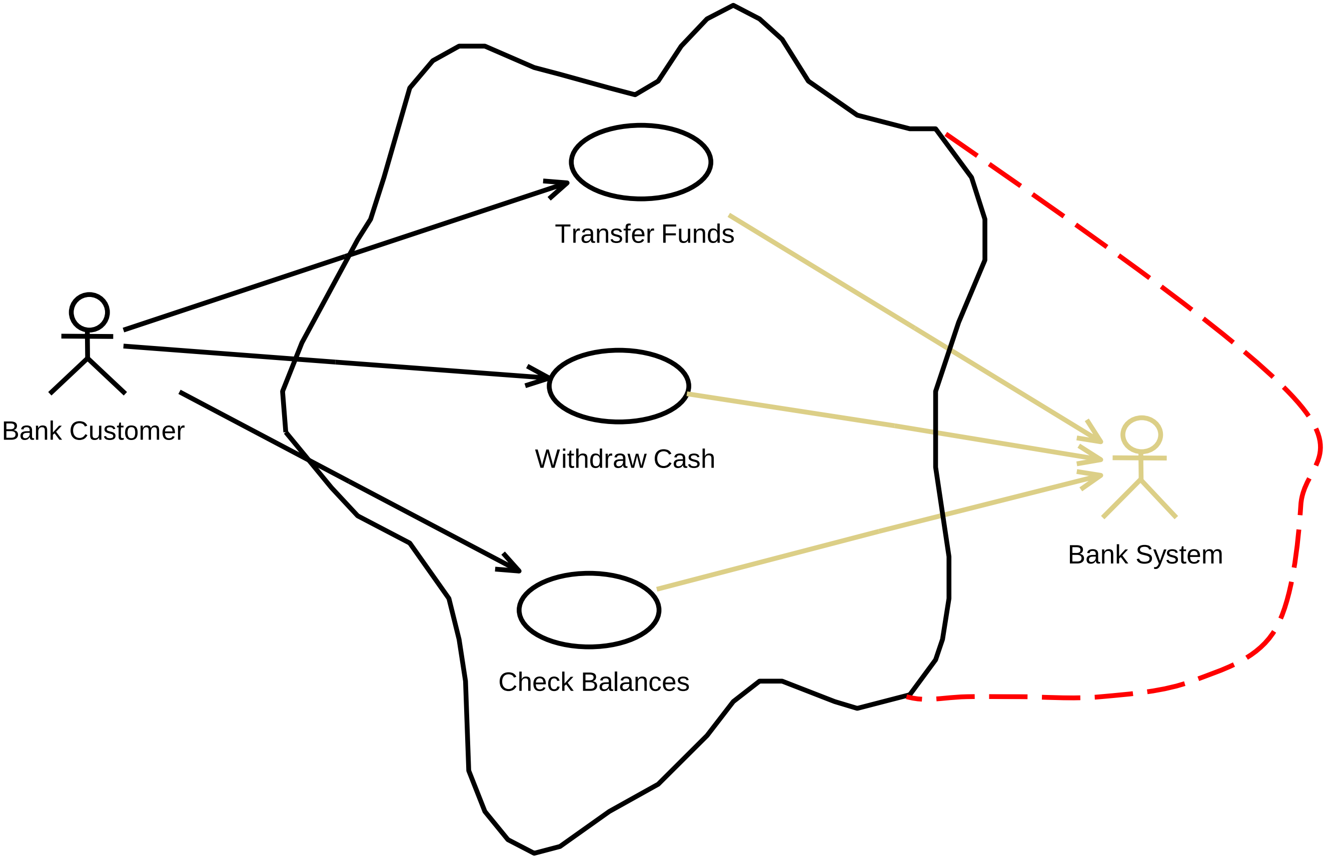
Will anyone else need to be informed of any events?

Remember the actors are not always people.

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Focus on the system boundary

- Considering other systems as actors forces you to confront the boundary of the system you are creating

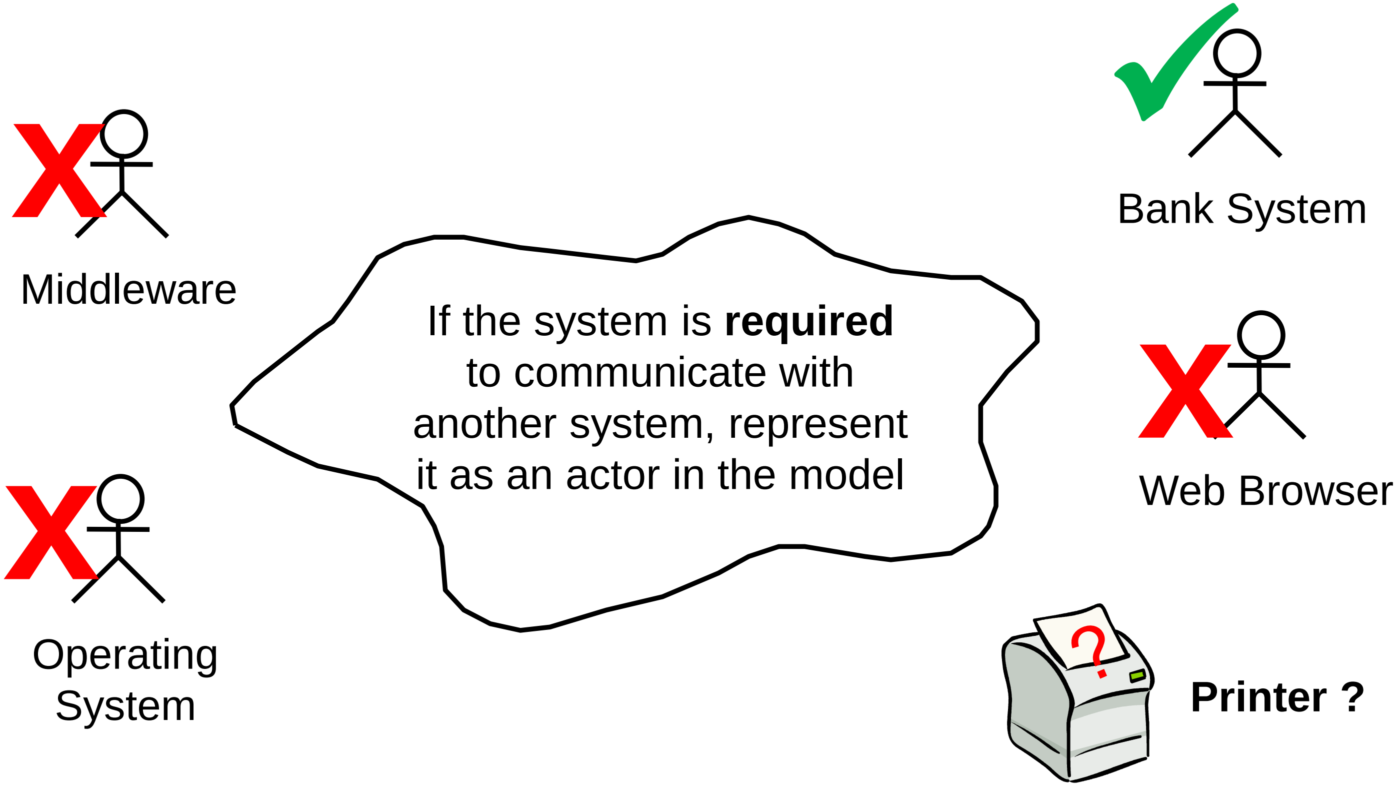


Where is the information that will be required to support the behavior?

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Discussion: Don’t pre-empt the design

- Is the other system really an actor or part of the system’s assembly?
- Is a printer an actor?
- What is the actor for batch processes?



If the system is **required** to communicate with another system, represent it as an actor in the model

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Naming your actors

- Actor names should describe the role the actor plays in relation to the system
- Good actor names are descriptive of their responsibilities

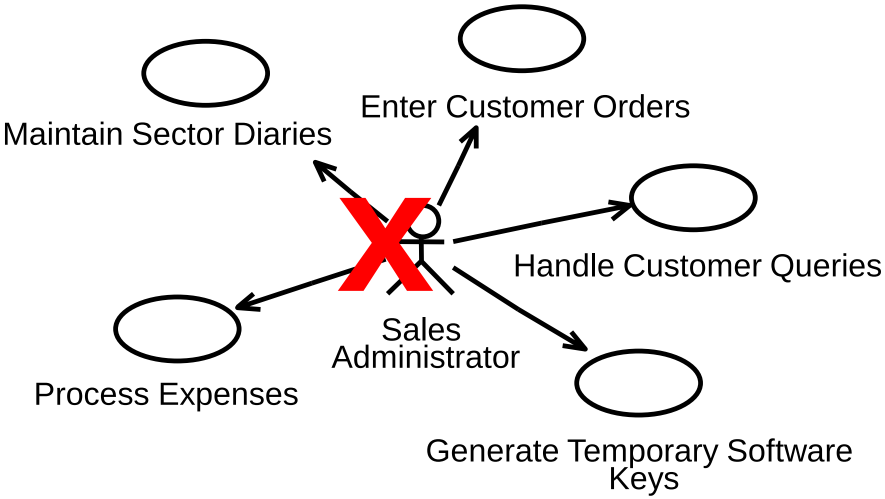
Actor Name	Comments
ATM Operator	Good actor name, describes the role of keeping the ATM in good working order and stocked with cash and paper.
Repair Person	Poor actor name, the actor’s role goes quite a bit beyond simply repairing the machine.

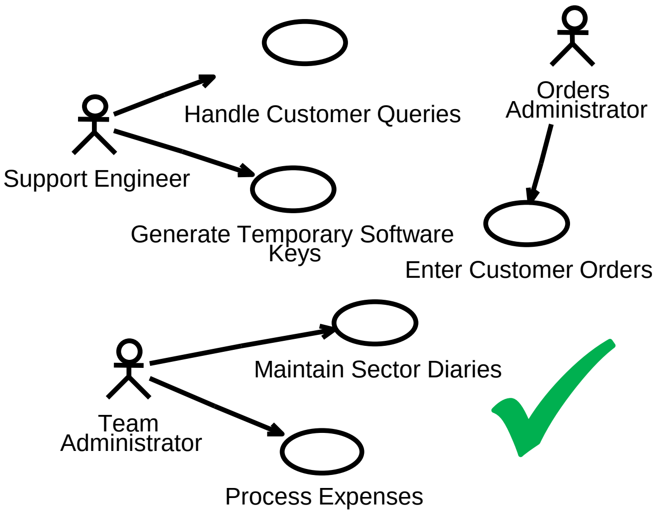
- Be sure to capture a brief description for each actor as soon as it is discovered
  - No more than a few sentences
  - Should describe the role the actor plays
  - Should state the goals the actor expects to achieve

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Actors – General rules

- Don’t confuse Actors with organizational roles/job titles
- Don’t use job titles in the Use-Case Model
- Don’t over generalize
- Use roles instead of job titles as Actor names
- Characterize your actors





Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

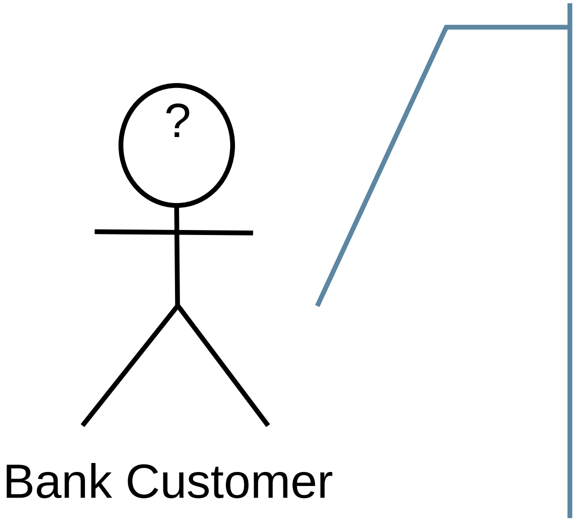
Exercise 2.2: Finding actors

- In your group:
  - Using the Model Storm results of the previous exercise
  - Name and Briefly Describe a set of Candidate Actors for the system to be developed
  - Choose someone to present your findings back to the class

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Finding use cases

- Start by identifying Actor Goals
- For each actor identified list the things that the actor needs to achieve by using the system



- Get an account balance from the ATM
- Make a cash Withdrawal from the ATM
- Request a statement from the ATM
- Deposit cash in the ATM. Etc……..

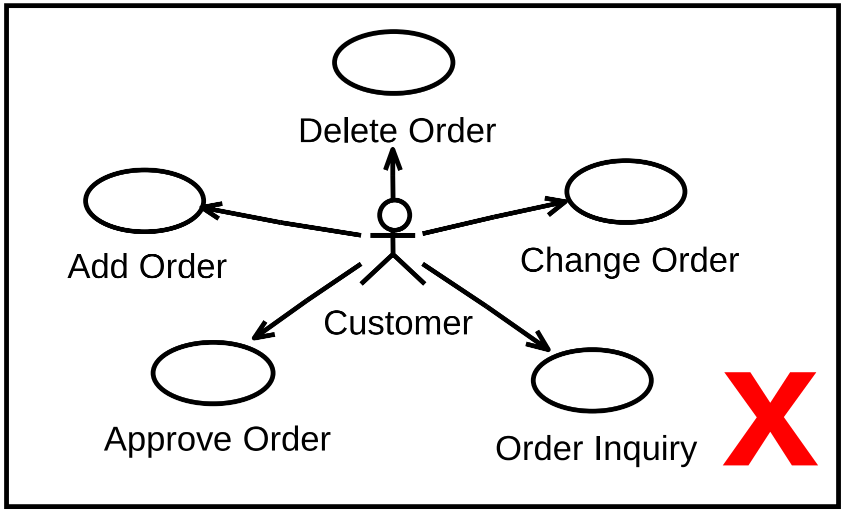
Candidate Use Cases

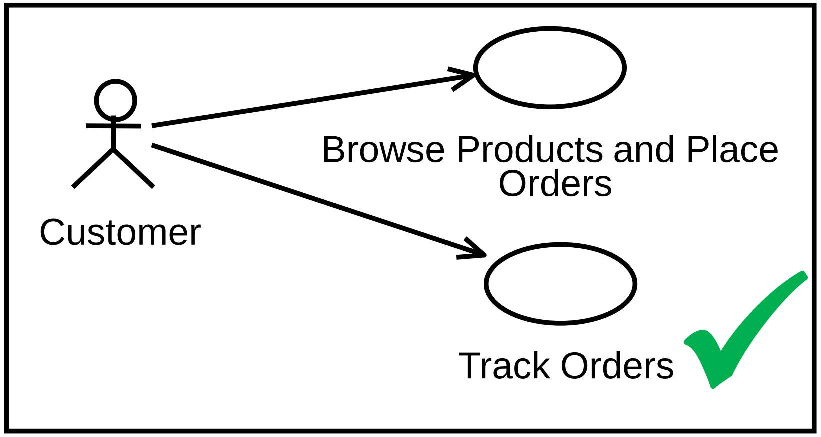
What about "Verify PIN"?

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Don’t confuse use cases with “Functions”

- Functional decomposition prevents the identification of meaningful scenarios and obscures the systems important requirements.





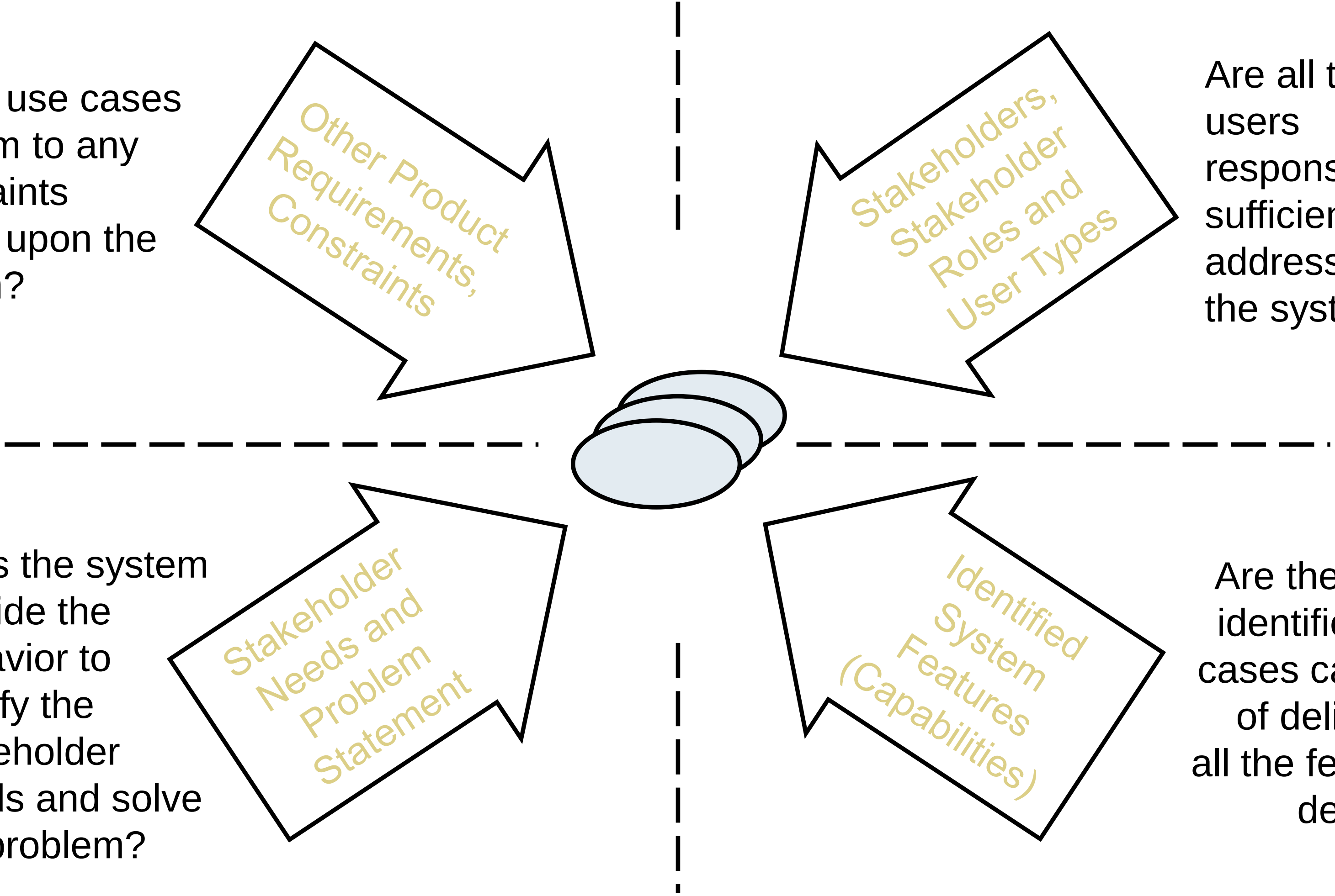
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Derive the use cases from the system's vision

- All existing requirements information should be leveraged to help identify candidate use cases

Do the use cases conform to any constraints placed upon the system?

Are all the users responsibilities sufficiently addressed by the system?


Other Product Requirements, Constraints
Stakeholders, Stakeholder Roles and User Types
Stakeholder Needs and Problem Statement
Identified System Features (Capabilities)

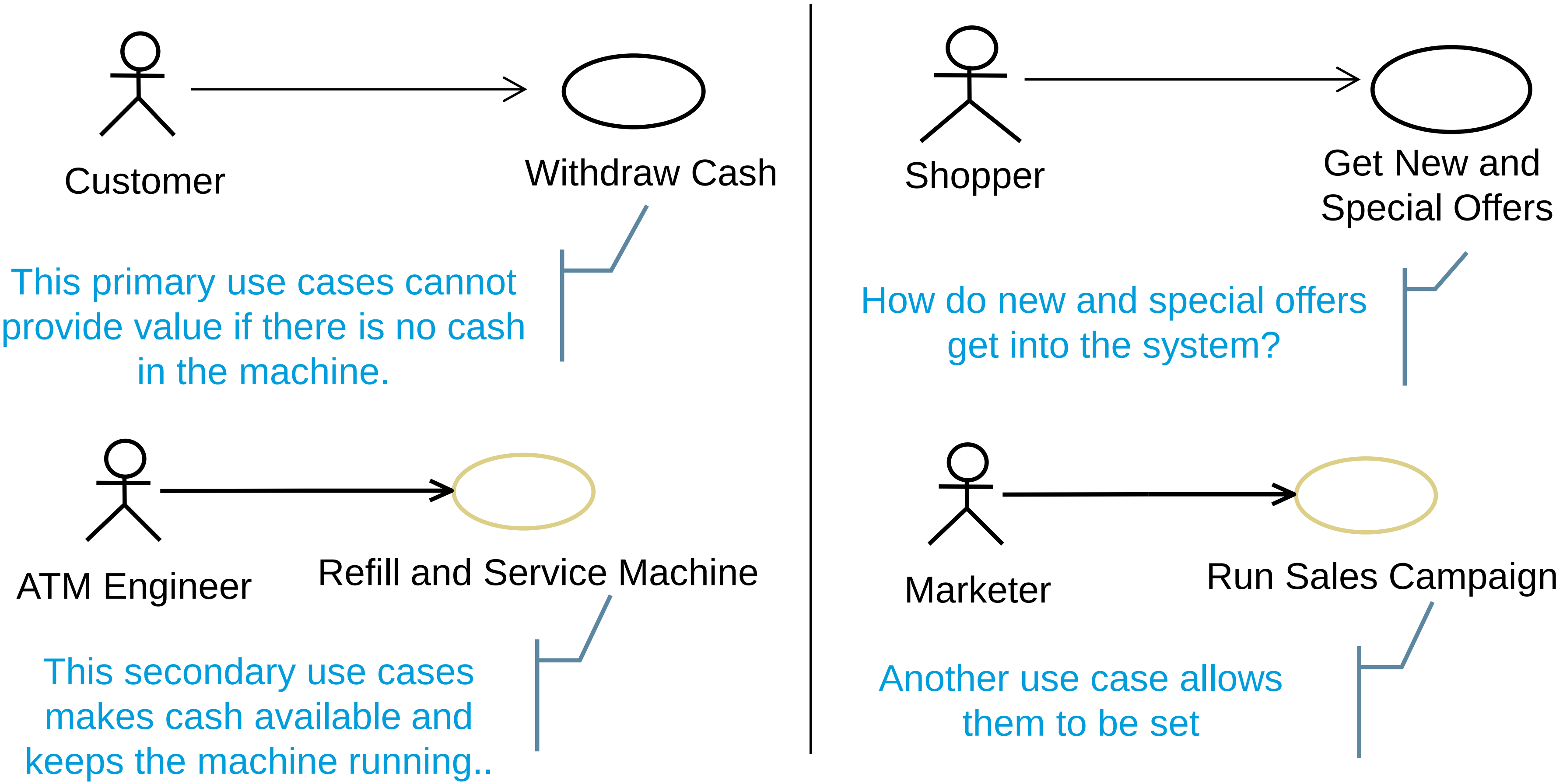
Does the system provide the behavior to satisfy the stakeholder needs and solve the problem?

Are the set of identified use cases capable of delivering all the features defined?

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Identify supporting and operational use cases

- Think about the information and other things that the actor will need to obtain from the system:
  - Does it get there by magic? How will it get there?



Customer

Withdraw Cash

This primary use cases cannot provide value if there is no cash in the machine.

ATM Engineer

Refill and Service Machine

This secondary use cases makes cash available and keeps the machine running..

Shopper

Get New and Special Offers

How do new and special offers get into the system?

Marketer

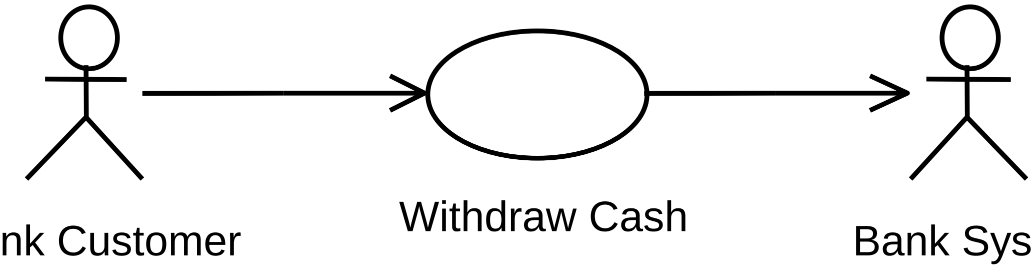
Run Sales Campaign

Another use case allows them to be set

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Associate the use cases to their actors

- After you have identified a use case with one or more associated actors:
  - Create a use-case diagram



- Remember:
  - The arrows indicate the initiator of the use case not “data flow”

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Capturing the requirements

- Evolve the set of use cases alongside the set of actors
  - The two activities go hand-in-hand, simultaneously and iteratively
  - Chicken–and–egg, the identification of new use cases can lead to the identification of more actors and vice-versa
- While identifying use cases, supporting information will also be identified
  - Glossary Terms, Business Rules, System Wide Quality Attributes, Standards, Regulations
  - System-wide requirements that don’t apply to any particular use case

The requirements should be evolved in parallel

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Name the use cases

- Give the use cases active names
  - Newly Identified Use Cases may have long names while brainstorming - this is a good start on the brief description

Passive Name	Active Name
Risk Assessment	Assess Risk
Flight Scheduling	Schedule Flight
Resource Management	Manage Resources

- Capture a brief description for each use case as soon as it is discovered
  - Never accept the argument that it is “obvious”; it seldom is
  - No more than a few sentences providing a short synopsis of what the system does to provide the value
  - Should make clear which Stakeholders are receiving the value
  - Should capture the specific value provided to those stakeholders

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Exercise 2.3: Finding use cases

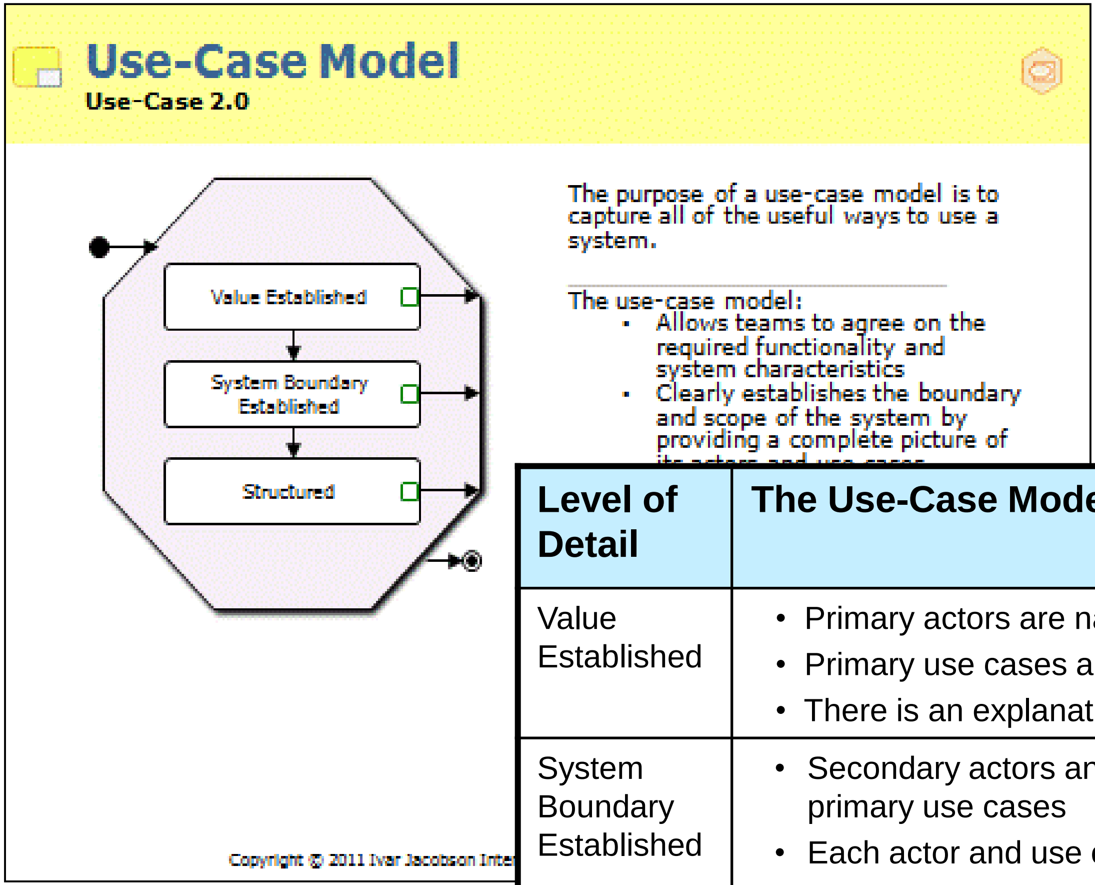
- In your group:
  - Using the Model Storm results and the Actors from the previous exercises
  - Name and Briefly Describe a set of Candidate Use Cases for the system to be developed
  - Create a first cut Use-Case diagram
  - Indicate which use cases address which features
  - Choose someone to present your findings back to the group

IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

How much do you need to model?



Use-Case Model

Use-Case 2.0

The purpose of a use-case model is to capture all of the useful ways to use a system.

The use-case model:

- Allows teams to agree on the required functionality and system characteristics
- Clearly establishes the boundary and scope of the system by providing a complete picture of its actors and use cases

Copyright © 2011 Ivar Jacobson Inte

Level of Detail	The Use-Case Model has achieved this level of detail when :
Value Established	• Primary actors are named and briefly described • Primary use cases are named and briefly described • There is an explanation of the use-case model as a whole
System Boundary Established	• Secondary actors and use cases have been identified to enable the primary use cases • Each actor and use case appears in at least one diagram • All relevant backlog items can be traced to use cases, actors or other requirements
Structured	• Include and generalization relationships are used to improve clarity of the use-case model. • Extends relationships are used to express extension behavior • The use-case model is still understandable to a novice stakeholder

IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

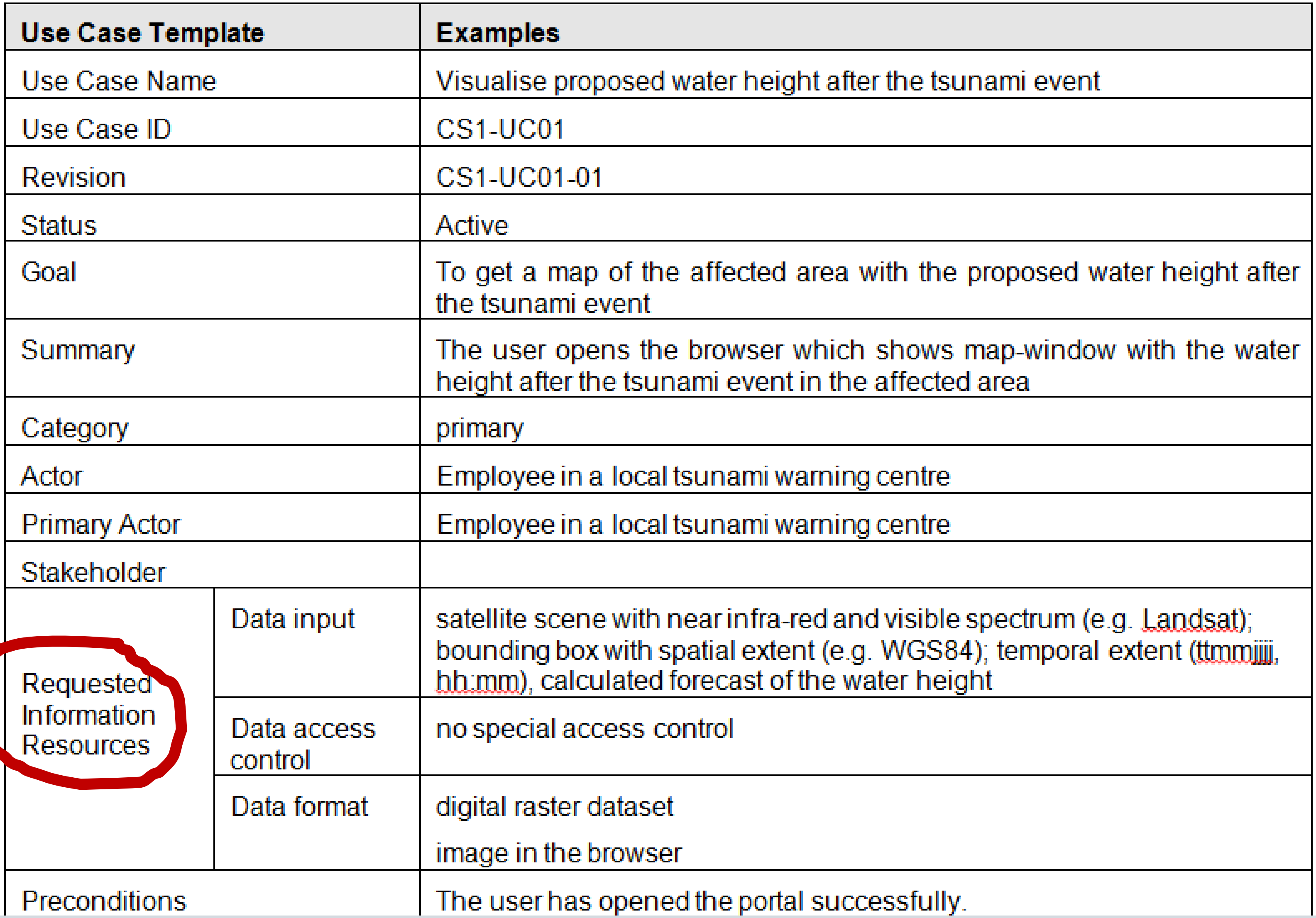
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Summary

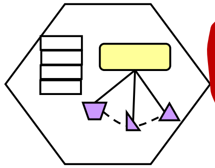
- The key to a successful start with use cases:
  - Understand the purpose and boundary of the system
  - When identifying Actors work from the specific to the general
  - Don't forget external systems that interact with the system being developed
  - A use case should provide independent value to the actor, if you have to execute several use cases in a sequence to add 'value', you have gone wrong
  - First focus on the obvious and familiar – don't get bogged down dealing with the unusual and uncommon
  - Evolve the set of actors and use cases alongside each other in an iterative and incremental fashion

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Template of a Use Case Description



Use Case Template		Examples
Use Case Name		Visualise proposed water height after the tsunami event
Use Case ID		CS1-UC01
Revision		CS1-UC01-01
Status		Active
Goal		To get a map of the affected area with the proposed water height after the tsunami event
Summary		The user opens the browser which shows map-window with the water height after the tsunami event in the affected area
Category		primary
Actor		Employee in a local tsunami warning centre
Primary Actor		Employee in a local tsunami warning centre
Stakeholder		
Requested Information Resources	Data input	satellite scene with near infra-red and visible spectrum (e.g. Landsat); bounding box with spatial extent (e.g. WGS84); temporal extent (ttmmjjjj, hh:mm), calculated forecast of the water height
	Data access control	no special access control
	Data format	digital raster dataset image in the browser
Preconditions		The user has opened the portal successfully.



Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Introduction to The Essentials

Module 3 – Use-Case Essentials

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Use-Case Essentials

- A way to establish the requirements of the system
  - Use cases place requirements in context
- A way to establish the system boundary
  - The model identifies who or what interacts with the system and what the system should do
- A way to iteratively evolve the requirements
- A way to communicate the requirements to all the stakeholders
  - The use cases provide a common thread through all project activities
- A way to focus the development efforts on delivering customer value
- A way to verify that the requirements have been implemented

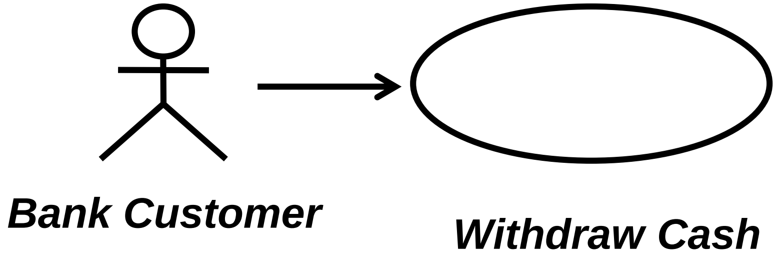
A way to effectively gather requirements and ensure that the system delivers real value to the customers and users

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

What is a Use Case?

A use case describes a sequence of actions a system performs that yields an observable *result of value* to a particular actor

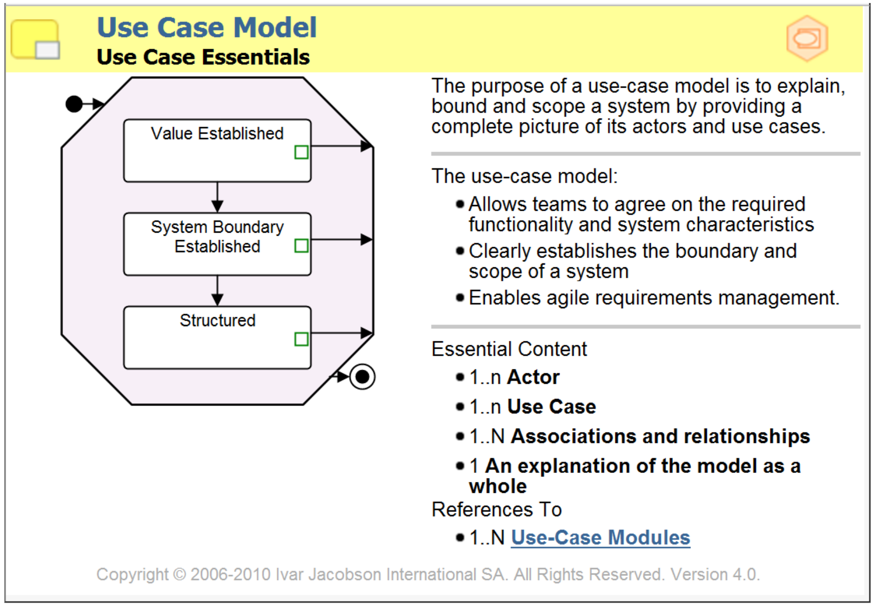
- Use cases are shown in UML diagrams



- Use cases are described in text
  - They tell the story of the interactions between actors and the system

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

What is a Use-Case Model?



Use Case Model

Use Case Essentials

The purpose of a use-case model is to explain, bound and scope a system by providing a complete picture of its actors and use cases.

The use-case model:

- Allows teams to agree on the required functionality and system characteristics
- Clearly establishes the boundary and scope of a system
- Enables agile requirements management.

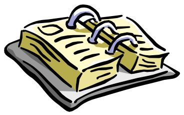
Essential Content

- 1..n **Actor**
- 1..n **Use Case**
- 1..N **Associations and relationships**
- 1 **An explanation of the model as a whole**

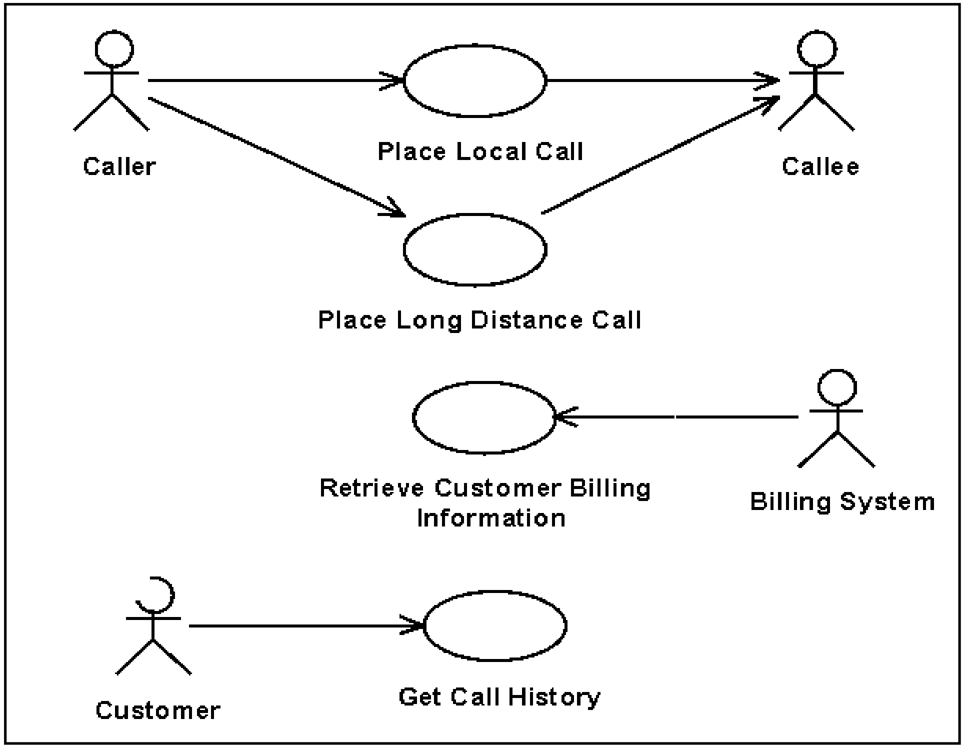
References To

- 1..N **Use-Case Modules**

Copyright © 2006-2010 Ivar Jacobson International SA. All Rights Reserved. Version 4.0.



Handout:
Example Use-Case Model



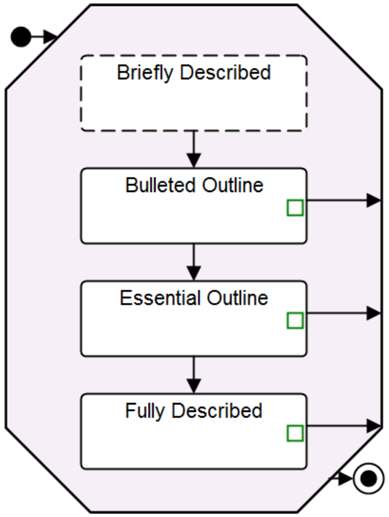
THE SMARTER WAY

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Describing a Use-Case

Use Case Specification

Use Case Essentials



A use-case specification describes how an actor uses a system to achieve a goal and what the system does for the actor to achieve that goal.

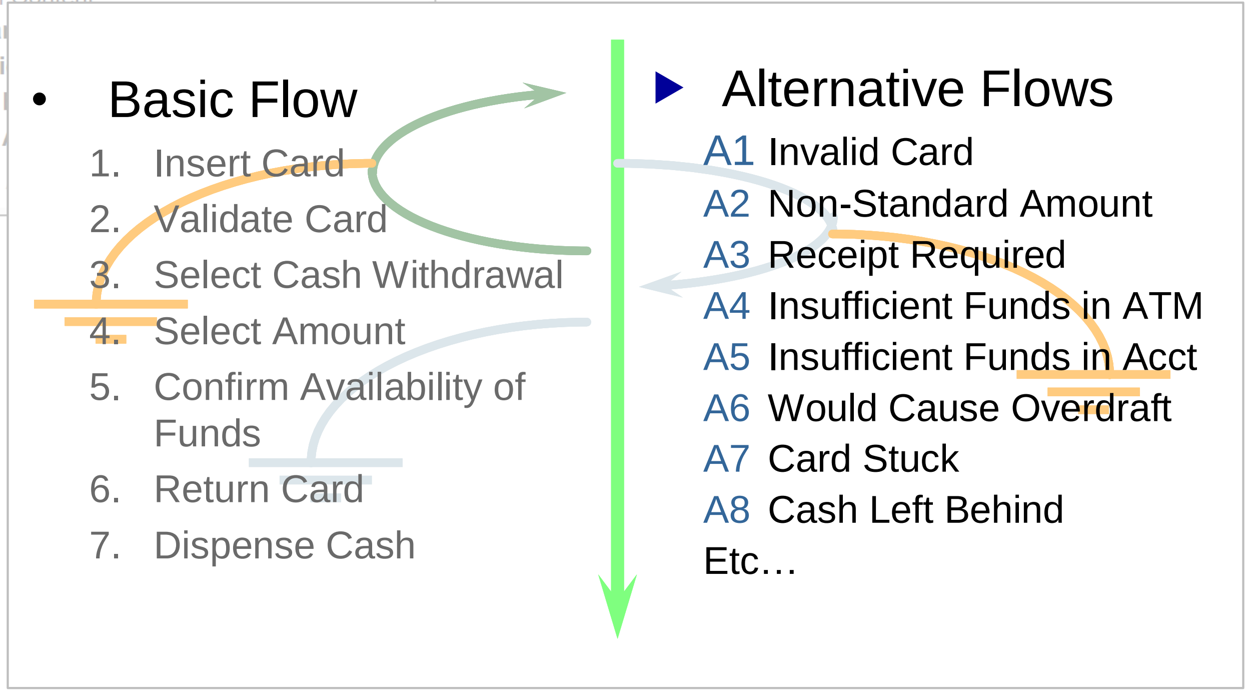
Use-case specifications:

- Capture requirements in context
- Define scenarios
- Enable effective scope management
- Provide the detail to drive the other development activities and ensure that they deliver value.

Essential Content

- 1 **Na**
- 1 **Bri**
- 1..N
- 0..N

Copyright © 2006-2010 Ivar Jacobson International SA.

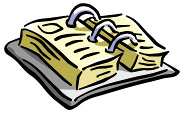


- Basic Flow
  1. Insert Card
  2. Validate Card
  3. Select Cash Withdrawal
  4. Select Amount
  5. Confirm Availability of Funds
  6. Return Card
  7. Dispense Cash

- Alternative Flows

  A1 Invalid Card
  A2 Non-Standard Amount
  A3 Receipt Required
  A4 Insufficient Funds in ATM
  A5 Insufficient Funds in Acct
  A6 Would Cause Overdraft
  A7 Card Stuck
  A8 Cash Left Behind
  Etc…

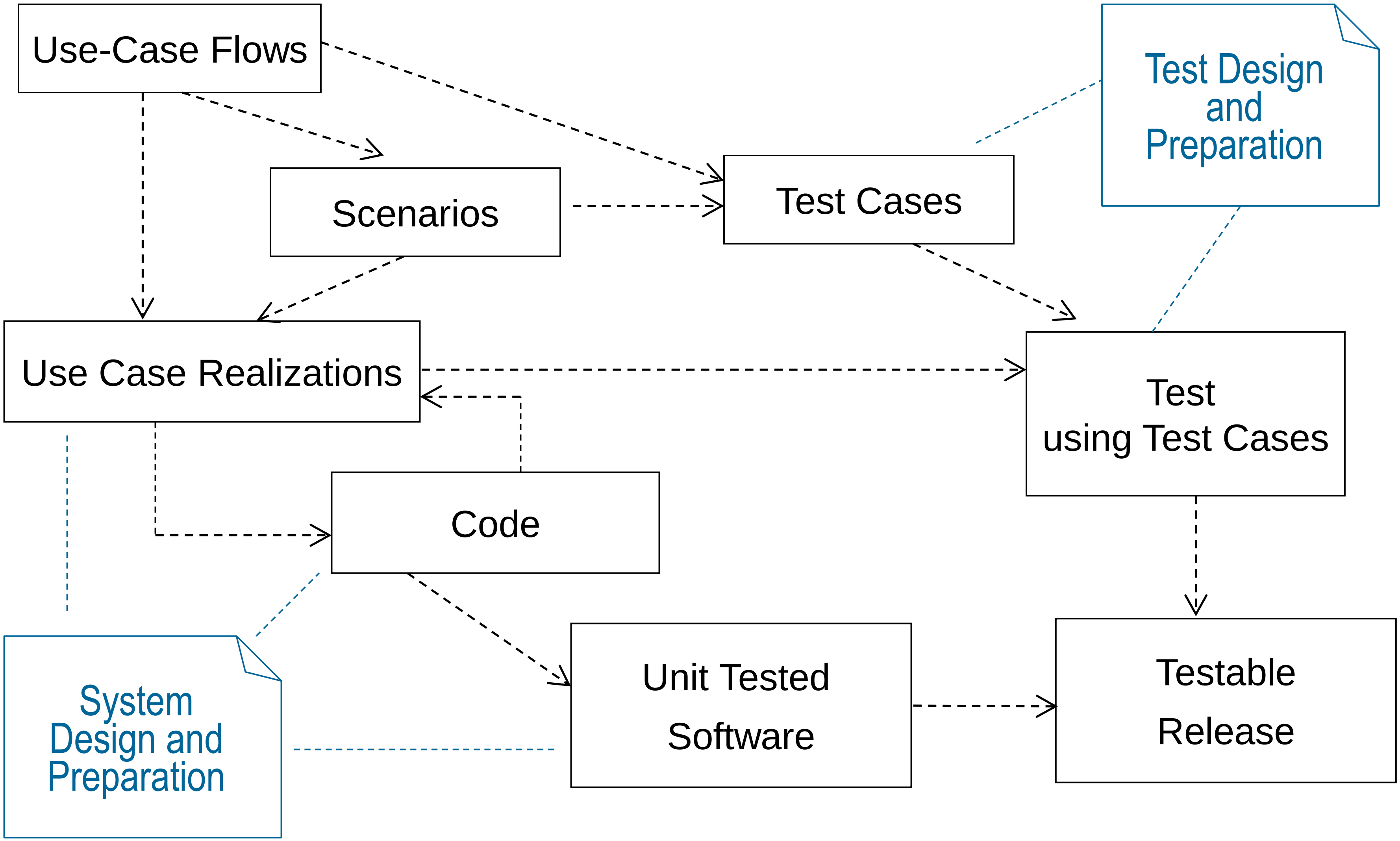
Handout: Example Use-Case (Outline)



Handout: Example Use-Case (Fully Described)

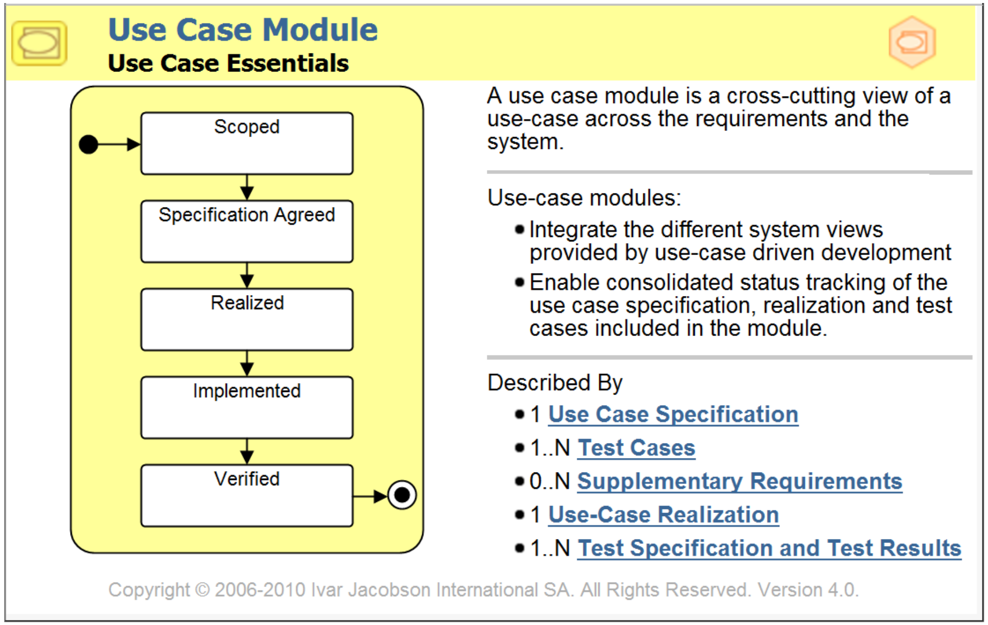
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Use Cases Drive Both Development and Testing



Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

The Use-Case Module



Use Case Module

Use Case Essentials

Scoped

Specification Agreed

Realized

Implemented

Verified

A use case module is a cross-cutting view of a use-case across the requirements and the system.

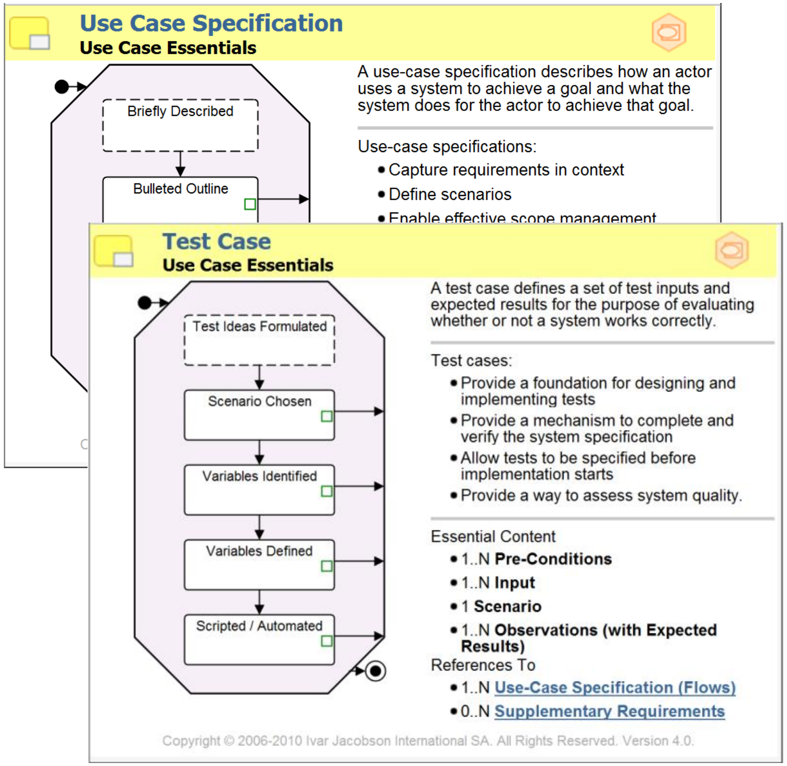
Use-case modules:

- Integrate the different system views provided by use-case driven development
- Enable consolidated status tracking of the use case specification, realization and test cases included in the module.

Described By

- 1 Use Case Specification
- 1..N Test Cases
- 0..N Supplementary Requirements
- 1 Use-Case Realization
- 1..N Test Specification and Test Results

Copyright © 2006-2010 Ivar Jacobson International SA. All Rights Reserved. Version 4.0.



Use Case Specification

Use Case Essentials

Briefly Described

Bulleted Outline

A use-case specification describes how an actor uses a system to achieve a goal and what the system does for the actor to achieve that goal.

Use-case specifications:

- Capture requirements in context
- Define scenarios
- Enable effective scope management

Test Case

Use Case Essentials

Test Ideas Formulated

Scenario Chosen

Variables Identified

Variables Defined

Scripted / Automated

A test case defines a set of test inputs and expected results for the purpose of evaluating whether or not a system works correctly.

Test cases:

- Provide a foundation for designing and implementing tests
- Provide a mechanism to complete and verify the system specification
- Allow tests to be specified before implementation starts
- Provide a way to assess system quality.

Essential Content

- 1..N **Pre-Conditions**
- 1..N **Input**
- 1 **Scenario**
- 1..N **Observations (with Expected Results)**

References To

- 1..N Use-Case Specification (Flows)
- 0..N Supplementary Requirements

Copyright © 2006-2010 Ivar Jacobson International SA. All Rights Reserved. Version 4.0.

The use-case module gathers together a set of use-case flows and their corresponding test cases to fully describe an aspect of the System

IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

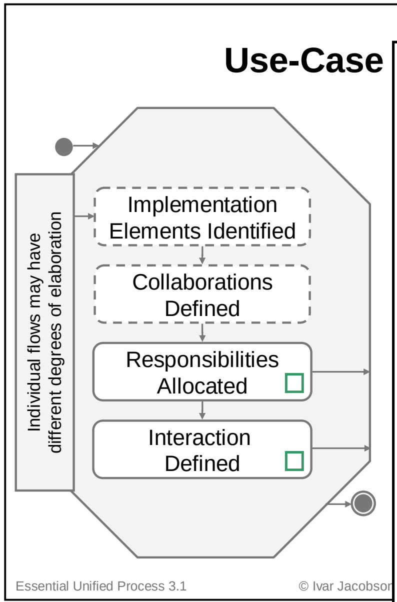
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

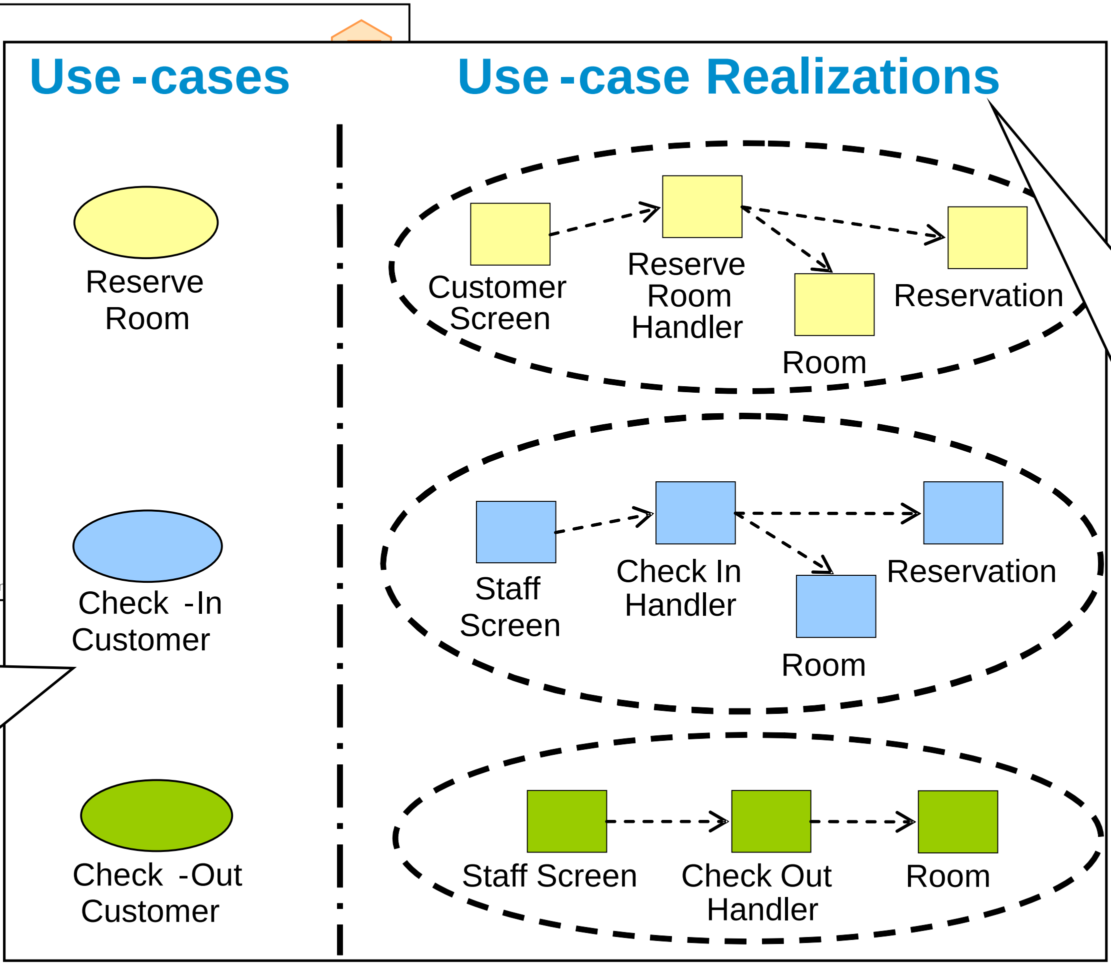
The use-case module is key to being agile

- Working with Use-Case Modules
  - Think about your risks and identify the key scenarios
  - Think about the natural groups of flows
  - Think about testing and proving the system
  - Think work items and about driving the development
- Lightweight use cases are enough to identify and reason about use-case modules
- Use-case modules are in a sense similar to user stories w.r.t. agility but there is a significant difference:
  - The context of a use-case model provides a more powerful way of reasoning about the requirements
    - Completeness/scope
    - Reference/detail/understandability
    - Scalability

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Use-case Realizations


Use-Case
Individual flows may have different degrees of elaboration
Implementation Elements Identified
Collaborations Defined
Responsibilities Allocated
Interaction Defined
Essential Unified Process 3.1
© Ivar Jacobson



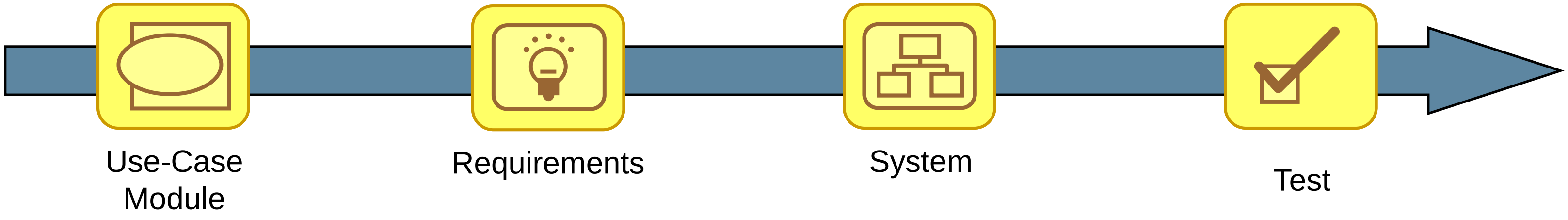
Use cases from the Specified System.

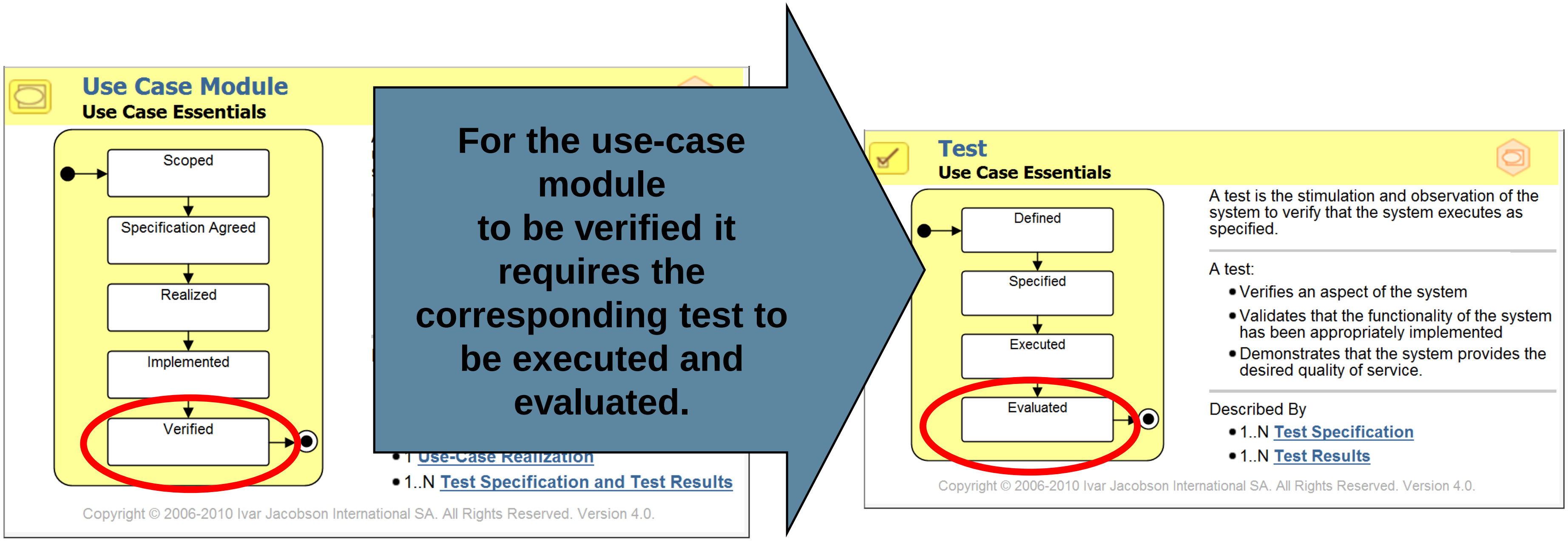
Elements of the Implemented System collaborating to perform the use cases.

Aligning the Specified and Implemented Systems

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Tests verify the implementation of the Use-Cases





Use Case Module

Use Case Essentials

- 1 Use-Case Realization
- 1..N Test Specification and Test Results

Copyright © 2006-2010 Ivar Jacobson International SA. All Rights Reserved. Version 4.0.

For the use-case module to be verified it requires the corresponding test to be executed and evaluated.

Test

Use Case Essentials

A test is the stimulation and observation of the system to verify that the system executes as specified.

A test:

- Verifies an aspect of the system
- Validates that the functionality of the system has been appropriately implemented
- Demonstrates that the system provides the desired quality of service.

Described By

- 1..N Test Specification
- 1..N Test Results

Copyright © 2006-2010 Ivar Jacobson International SA. All Rights Reserved. Version 4.0.

IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

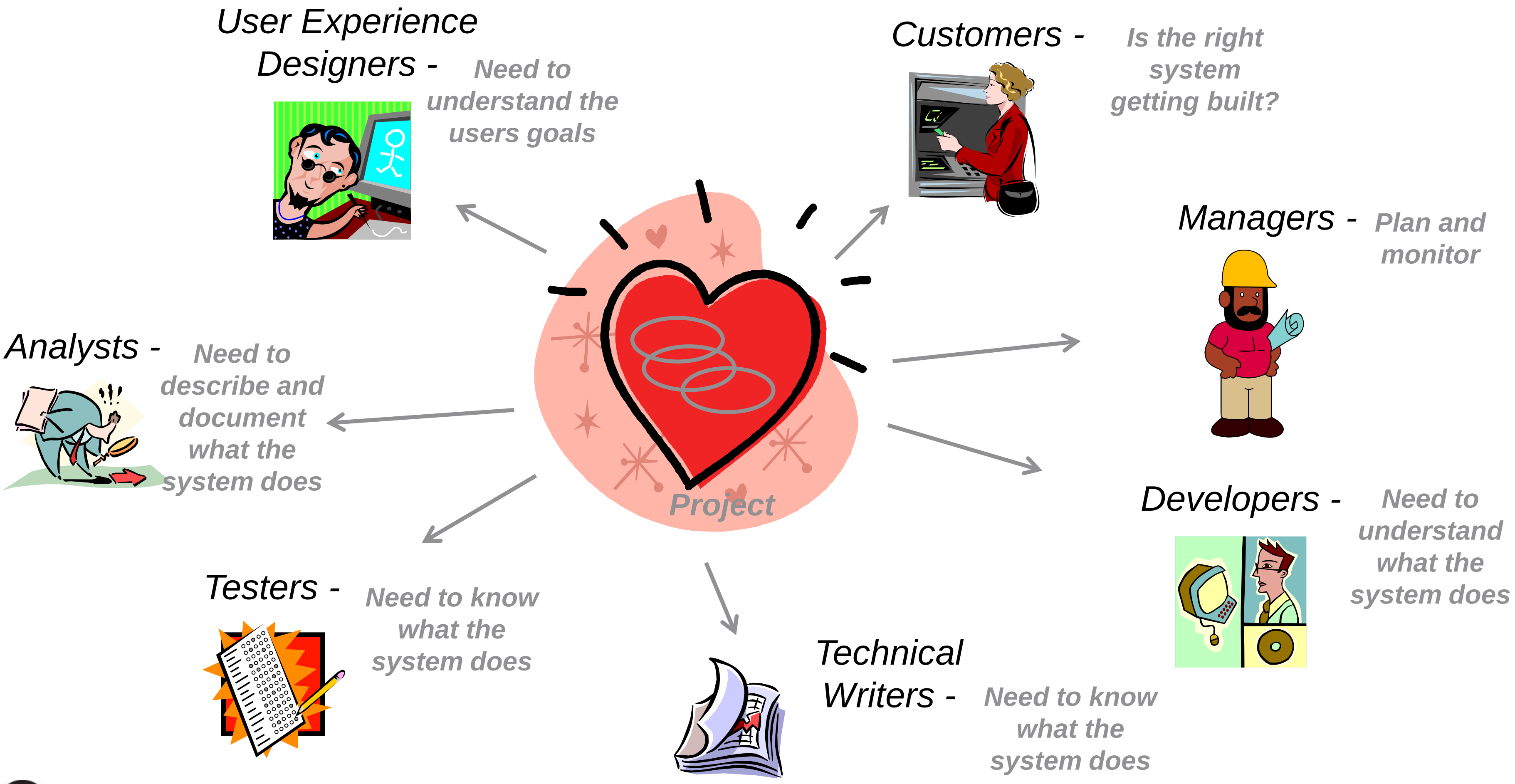
Use-Case Specifications enable agility

Authoring State	Primary Purpose	Supports
Briefly Described	Identify the use case and summarize its purpose.	• Basic scope management • Discussions about requirements
Bulleted Outline	Summarize the shape and extent of the use case.	• Scope management • Low fidelity estimation. • Collaborative test definition • Impact analysis and prototyping. • Component identification
Essential Outline	Summarize the essence of the use case.	• User Interface design. • Prototyping. • Collaborative, creative analysis and design • Collaborative test definition • High fidelity estimation
Fully Described	Provide a full, detailed requirements specification for the use case.	• Analysis and design • Implementation and testing • Creation of user documentation. • High fidelity estimation

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Use Cases and the project team

- Use Cases can be used as a unifying principle that unites the activities of the project



User Experience Designers - ***Need to understand the users goals***

Customers - ***Is the right system getting built?***

Managers - ***Plan and monitor***

Analysts - ***Need to describe and document what the system does***

Project

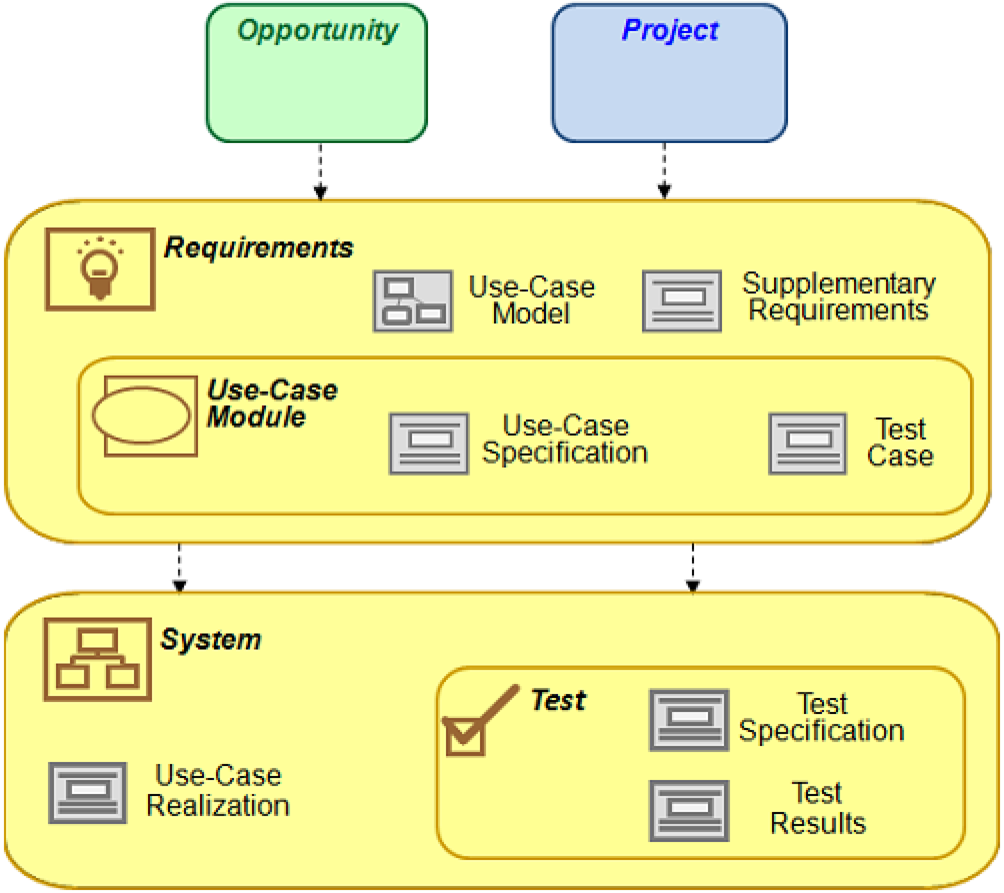
Developers - ***Need to understand what the system does***

Testers - ***Need to know what the system does***

Technical Writers - ***Need to know what the system does***

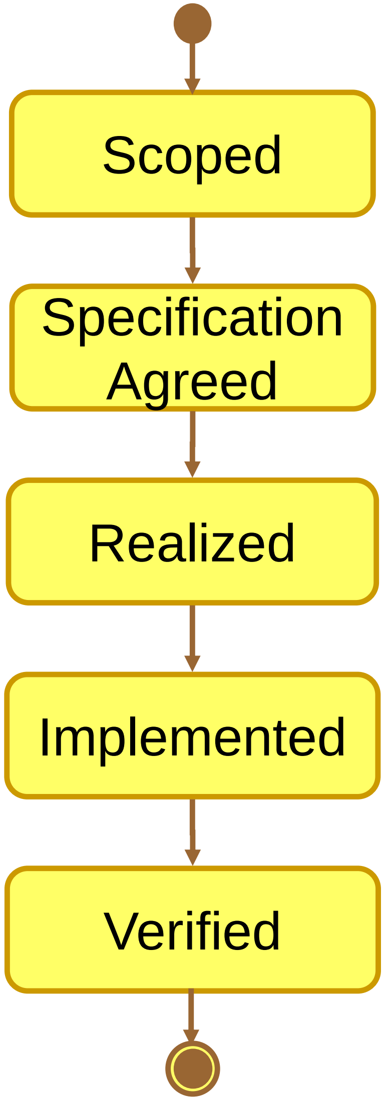
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

What do we need to produce?



Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

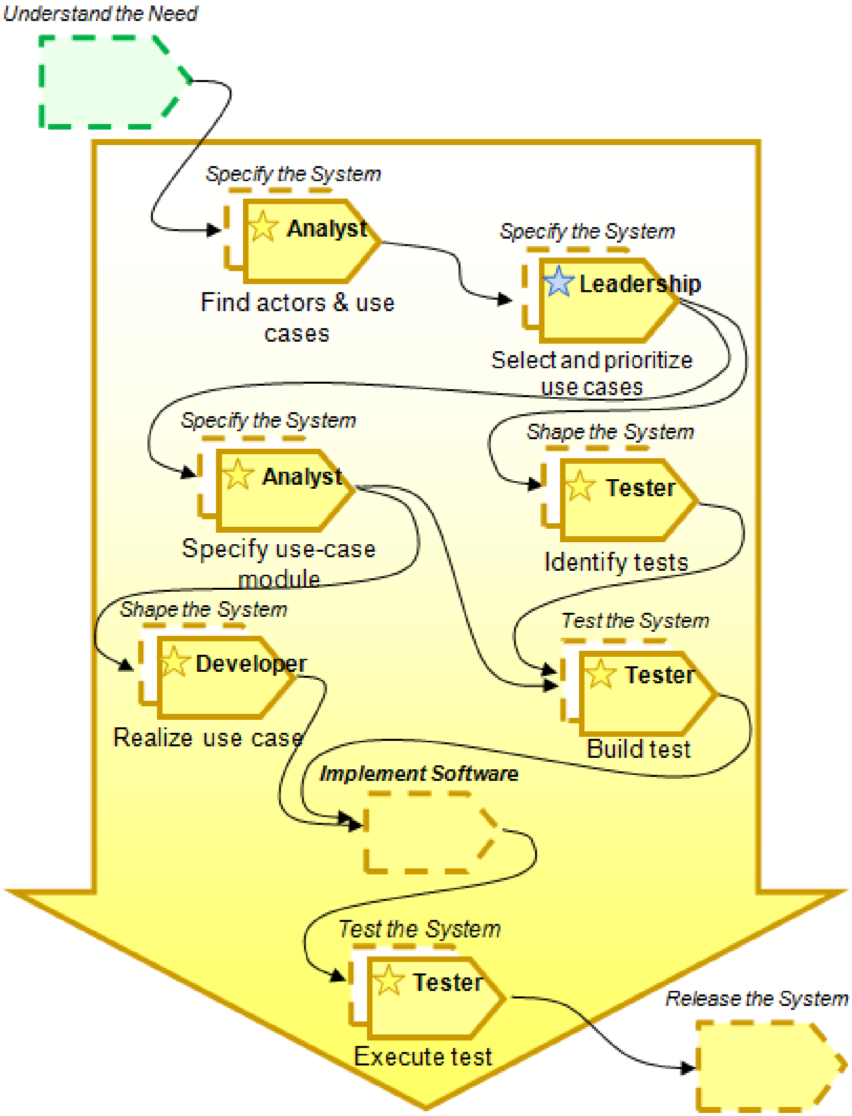
How do we progress and complete the use-cases?



- Scope the use-case module
  - Identify the flows to be described and implemented
- Agree the Specification
  - Write the use-case specification
  - Write the test cases
  - Remember to cover special and supplementary requirements
- Realize the flows of events
  - Allocate the requirements described by the flow of events to the elements of the Implemented System
  - Understand the impact of implementing the new requirements
- Implement software to deliver the use case
  - Amend the affected implementation elements
  - Integrate the system
- Verify the system produced
  - Execute tests based on the test cases to verify the system delivers the use cases as specified.

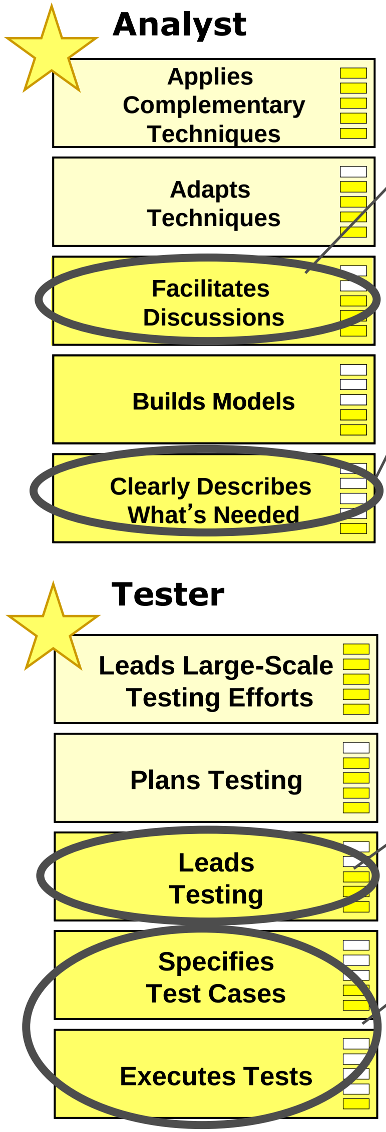
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

What activities do we need to do?

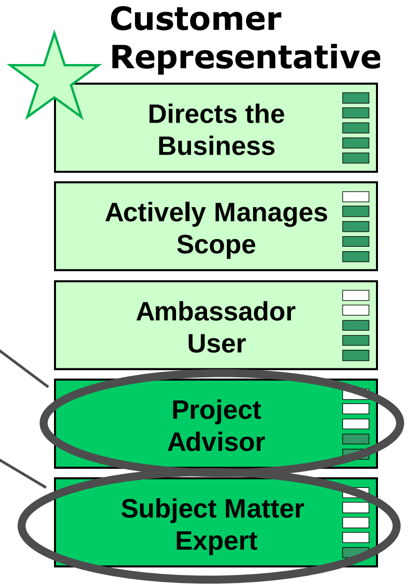


Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

What competencies do we need?



- At least one team member able to:
  - Build the use-case model and facilitate workshops
- Many team members able to:
  - Specify the use cases
  - Involve stakeholders so that true requirements are captured
- At least one customer able to:
  - Advise the project
- Many customers to be able to:
  - Provide information
- At least one team member able to:
  - Identify an appropriate set of tests
- Many team members to:
  - Specify test cases
  - Involve stakeholders so that true requirements are tested
  - Prepare the executable system for testing



IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

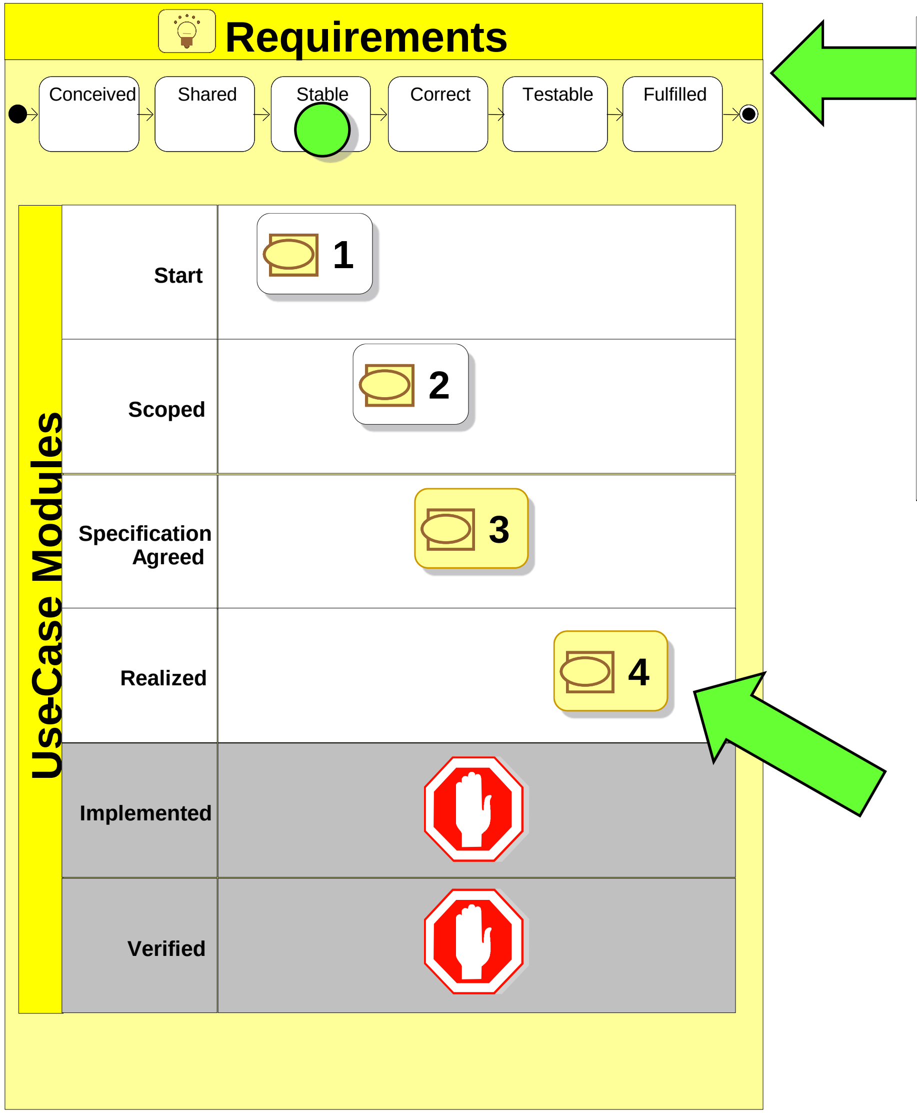
Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

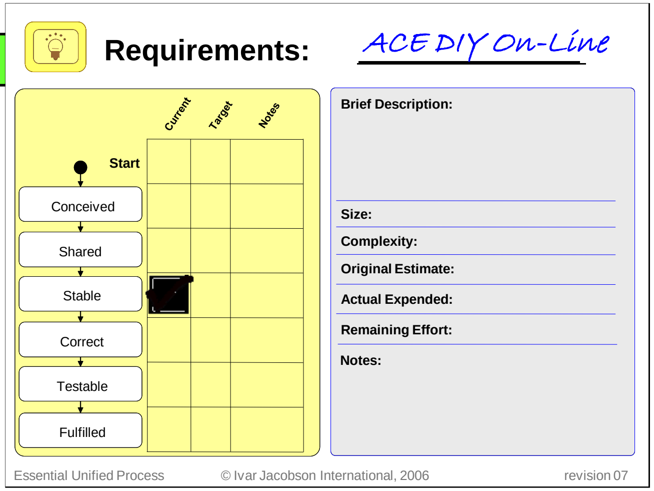
Some important Do’s and Don’ts

- Do track progress with use cases
  - Track progress in terms of verified requirements
  - The software does not count as finished until it has been verified against the use case
- Do select use-cases to implement based on project risk
  - The flows in the use case allow specific risks to be targeted
- Do use use-case modules to enable scenario-based planning
  - The use-case modules and their flows provide an effective scenario-based planning mechanisms
- Do just-enough requirements
  - Stop at the first level of detail that addresses your risks
- Do share use cases with the customers
  - Customer collaboration is essential to produce good use-case models and use-case specifications

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

The use-case modules control the game





Requirements: *ACE DIY On-Line*

	Current	Target	Notes
Start			
Conceived			
Shared			
Stable	■		
Correct			
Testable			
Fulfilled			

Brief Description:

Size:

Complexity:

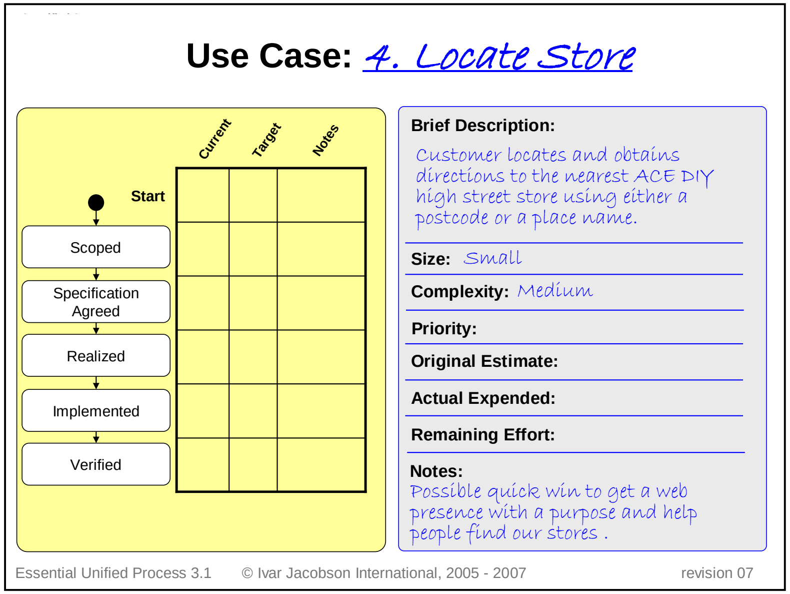
Original Estimate:

Actual Expended:

Remaining Effort:

Notes:

Essential Unified Process © Ivar Jacobson International, 2006 revision 07



Use Case: *4. Locate Store*

	Current	Target	Notes
Start			
Scoped			
Specification Agreed			
Realized			
Implemented			
Verified			

Brief Description:
Customer locates and obtains directions to the nearest ACE DIY high street store using either a postcode or a place name.

Size: *Small*

Complexity: *Medium*

Priority:

Original Estimate:

Actual Expended:

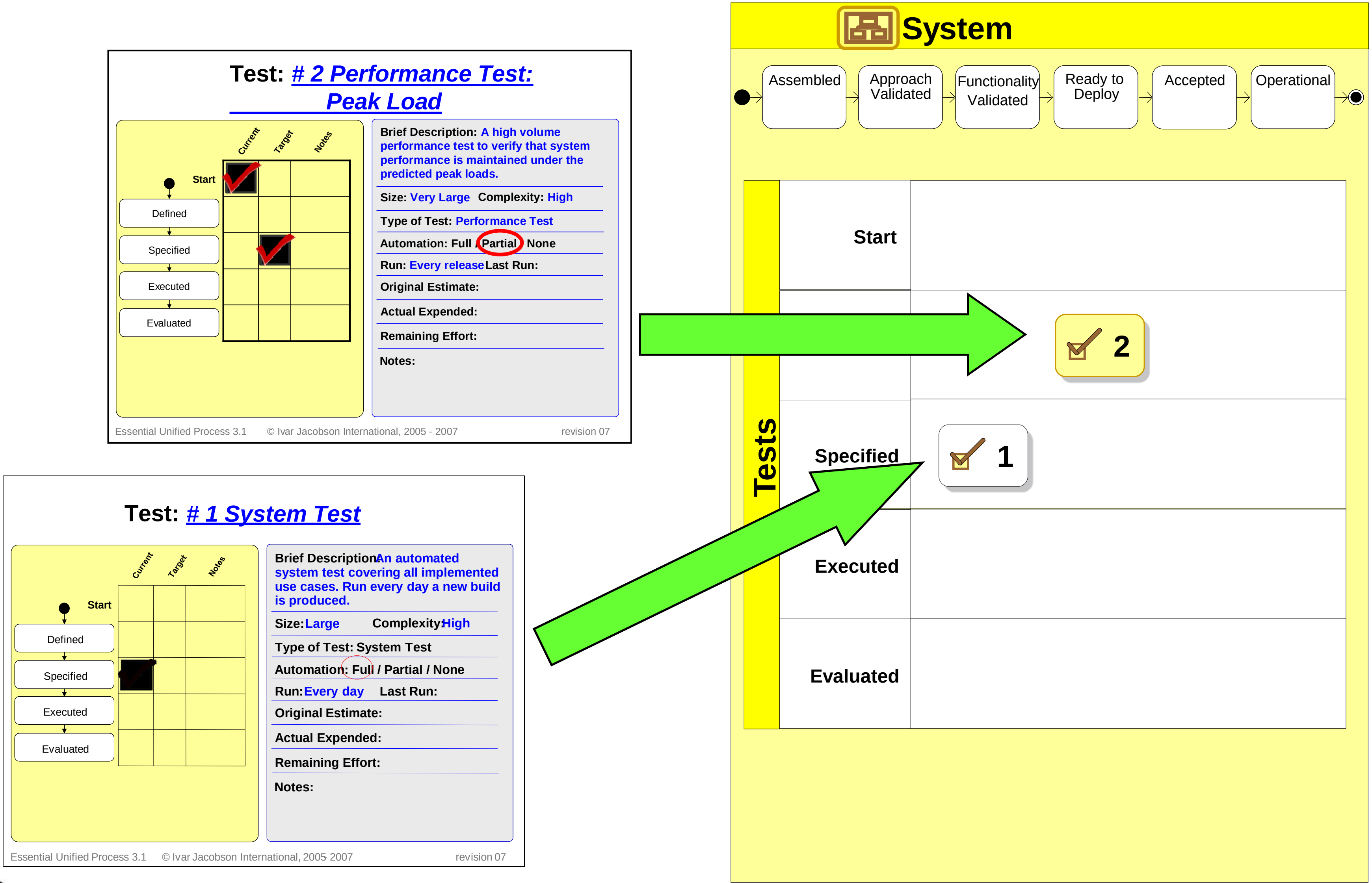
Remaining Effort:

Notes:
Possible quick win to get a web presence with a purpose and help people find our stores .

Essential Unified Process 3.1 © Ivar Jacobson International, 2005 - 2007 revision 07

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Tests are needed to verify the implementation



Test: # 2 Performance Test: Peak Load

	Current	Target	Notes
Start	✓		
Defined			
Specified		✓	
Executed			
Evaluated			

Brief Description: A high volume performance test to verify that system performance is maintained under the predicted peak loads.

Size: Very Large Complexity: High

Type of Test: Performance Test

Automation: Full / [✓ Partial] None

Run: Every release Last Run:

Original Estimate:

Actual Expended:

Remaining Effort:

Notes:

Essential Unified Process 3.1 © Ivar Jacobson International, 2005 - 2007 revision 07

Test: # 1 System Test

	Current	Target	Notes
Start			
Defined			
Specified	✓		
Executed			
Evaluated			

Brief DescriptionAn automated system test covering all implemented use cases. Run every day a new build is produced.

Size:Large Complexity:High

Type of Test: System Test

Automation: [✓ Full] / Partial / None

Run:Every day Last Run:

Original Estimate:

Actual Expended:

Remaining Effort:

Notes:

Essential Unified Process 3.1 © Ivar Jacobson International, 2005 2007 revision 07

IVAR JACOBSON INTERNATIONAL

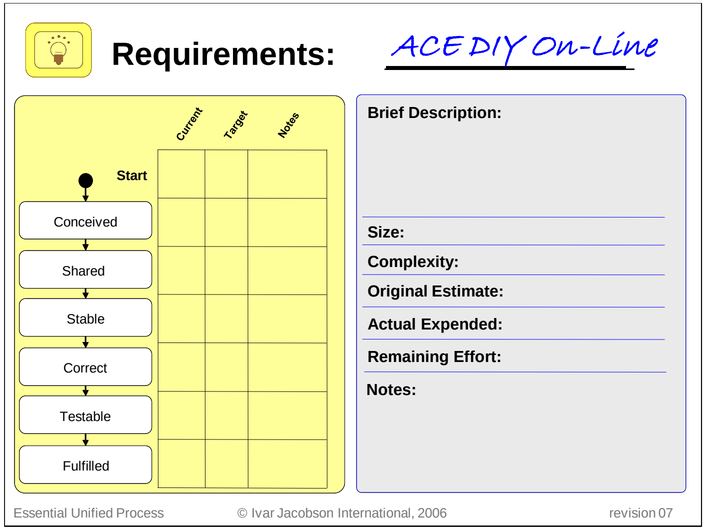
THE SMARTER WAY

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Exercise: Applying the use-case essentials practice

Using:

- The exercise instructions
- The supplied instance cards and matrix
- The supplied game boards



Requirements: ACE DIY On-Line

	Current	Target	Notes
Start			
Conceived			
Shared			
Stable			
Correct			
Testable			
Fulfilled			

Brief Description:

Size:

Complexity:

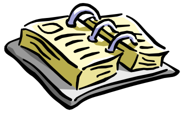
Original Estimate:

Actual Expended:

Remaining Effort:

Notes:

Essential Unified Process © Ivar Jacobson International, 2006 revision 07



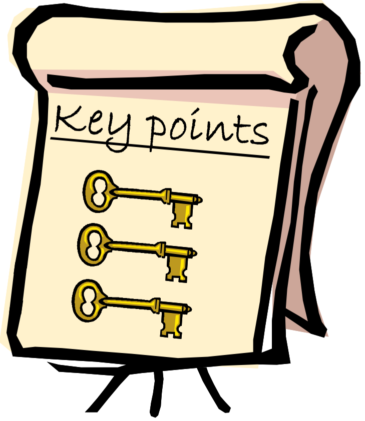
Handout:
State to activity matrix

Apply the use-case essentials practice to simulate the development of the Ace DIY system.

Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved

Summary

- The Use-Case Essentials practice enables teams to specify and drive the development of a complex software solution
  - It involves peopled skilled in analysis and testing, and with the ability to effectively represent the customers
  - It captures the requirements in the form of use cases and described how to realize and test these use cases to ensure that a high quality, high value system is produced
  - It can be used with any software development practices
  - It supports many models of team size and distribution including
    - Small collocated teams
    - Large distributed teams
    - Out-sourced development activities
- Other practices may be needed to
  - Define the business case and project plans
  - Develop the software
  - Release the system



Copyright © 2006-2010 Ivar Jacobson International SA. All rights reserved