IVAR JACOBSON INTERNATIONAL

THE SMARTER WAY

Use-Case 2.0: Scaling up, scaling out, scaling in for agile projects

Kurt Bittner
CTO - Americas

ivar@ivarjacobson.com

www.ivarjacobson.com

Scaling up, scaling out, scaling in – what is that?

Use cases scale in several dimensions:

- ***Scaling up***:
  - Though Use-Case 2.0 is designed for small teams and small projects, it scales without changing the fundamentals to large organizations and large projects.
- ***Scaling out***:
  - Though Use-Case 2.0 starts with requirements, it scales to many other lifecycle activities such as analysis, design, code, test, user experience, business design, etc.
- ***Scaling in***:
  - Use-Case 2.0 allow you to be as light as want, focusing on the essentials only, or to scale with more and more detail for systems such as telecom or defense systems or more regulated systems such as life-critical systems.

Still with the constraint that you work with agile principles and values

Agenda

- A Brief History of Use Cases
- Use-Case 2.0 – What is new?
- Use-Case 2.0 in practice
- Wrap Up

A brief history of use cases

Past

‘87 – ‘96

Use-Case Driven Development Paper, OOPSLA, 87

The Objectory Process and Object-Oriented Software Engineering, Addison Wesley, 1992

Object-Oriented Software Engineering
A Use Case Driven Approach

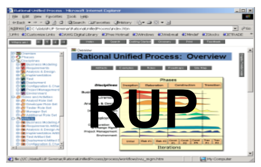
UML, OOPSLA, 1995

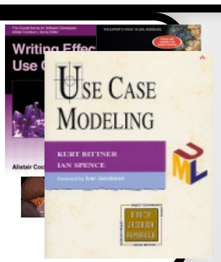
’96 -’00

The Rational Objectory Process, 1997

The Unified Software Development Process, Addison Wesley, 1999

More and more books and methods using use cases.



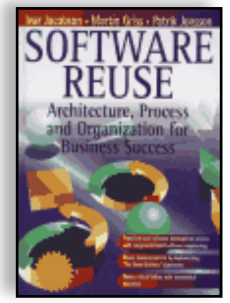
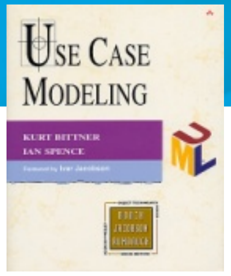
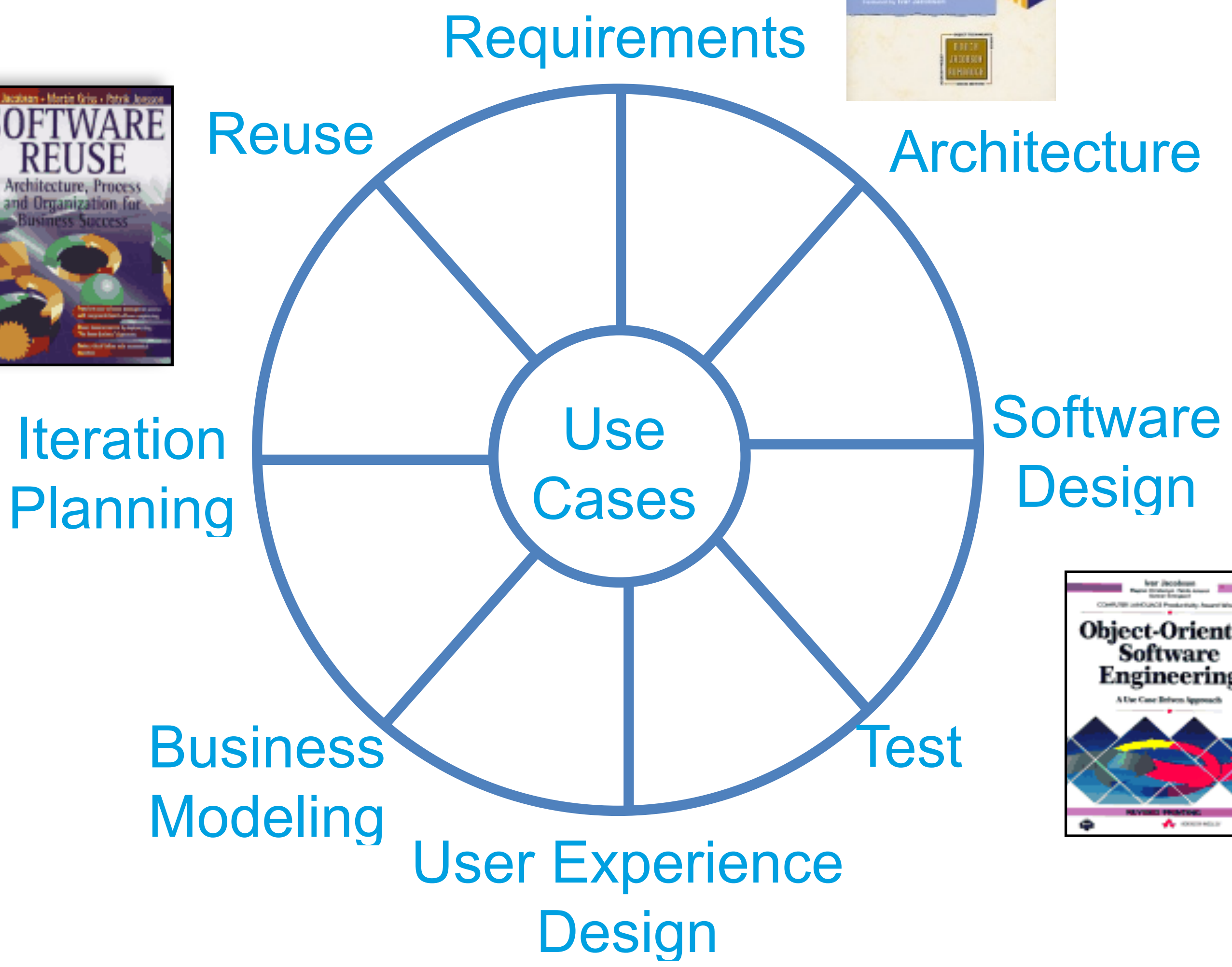


Present

And now Use-Case 2.0…

What made use cases so popular?

- They effectively communicate what a system is supposed to do
- They place the requirements into the context of a specific user's goals
- They are the test cases
- They are the starting point for the design of effective user experiences
- They 'drive' the development through design and code

Use-Case Modeling – A very simple idea.

To get to the heart of what a system must do, you should focus on who, (or what) will use it, and then look at what the system must do for them to help them achieve something useful.

IVAR JACOBSON INTERNATIONAL

Why do we still need use cases?

"There are lots of other popular requirement related practices, haven't they replaced the need for use cases?"

- User Stories – great for small systems and small teams
- Features – great for product management
- Declarative Requirements – great for capturing independent, atomic qualities
- Domain Modeling – great for information rich, functionally simple systems

There a lots of great techniques but they lack something to pull them together, and provide a simple, scalable solution.

Why do we need Use Case 2.0?

- To correct some of the common misunderstandings:
  - Use-cases are lightweight **not** heavy-weight
  - Use-cases are stories **not** functions
  - Use-cases are simple **not** complicated
  - Use-cases are for all types of development **not** just green field application development
- To re-focus on the essentials
- To better support innovations and improvements such as test-driven development, Kanban, and Scrum

Use-Case 2.0

Scaling up, scaling out, scaling in.
The lightness of user stories with the power of modeling.

Agenda

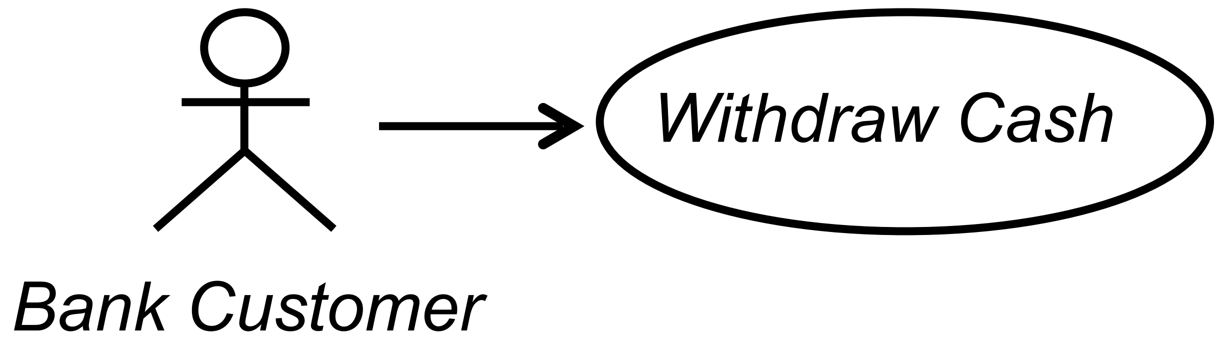
- A Brief History of Use Cases
- Use-Case 2.0 – What is new?
- Use-Case 2.0 in practice
- Wrap Up

IVAR JACOBSON INTERNATIONAL

A use case is still a use case

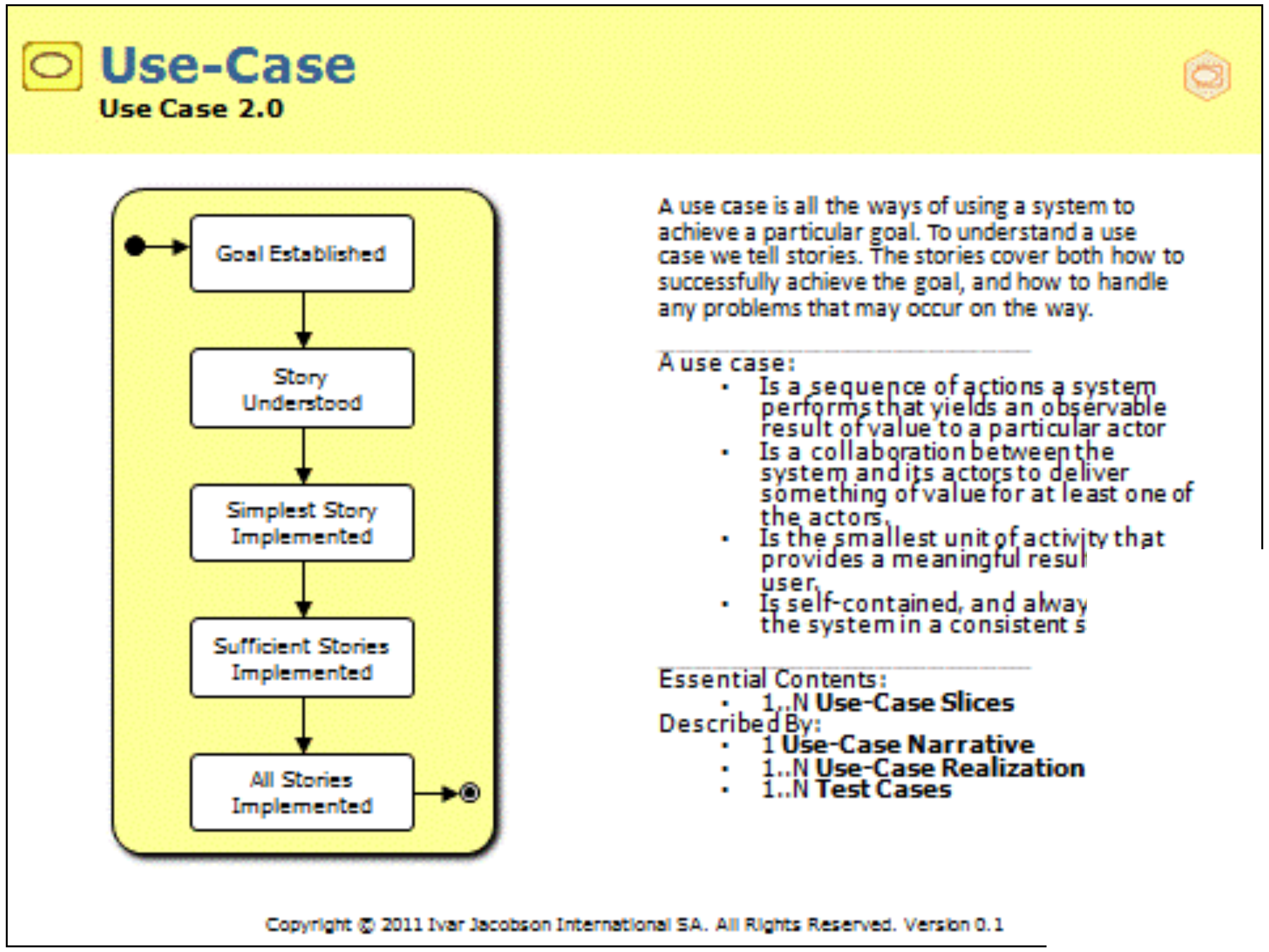
A use case is all the ways of using a system to achieve a particular goal for a particular user.

- Use cases can be shown in UML diagrams
- Use cases are described as narratives
  - Which tell the story of how the system and its users work together to achieve a particular goal



…but the way we describe and apply them has evolved

Use cases act as placeholders for conversations


Goal Established
Story Understood
Simplest Story Implemented
Sufficient Stories Implemented
All Stories Implemented

Use-Case
Use Case 2.0

A use case is all the ways of using a system to achieve a particular goal. To understand a use case we tell stories. The stories cover both how to successfully achieve the goal, and how to handle any problems that may occur on the way.

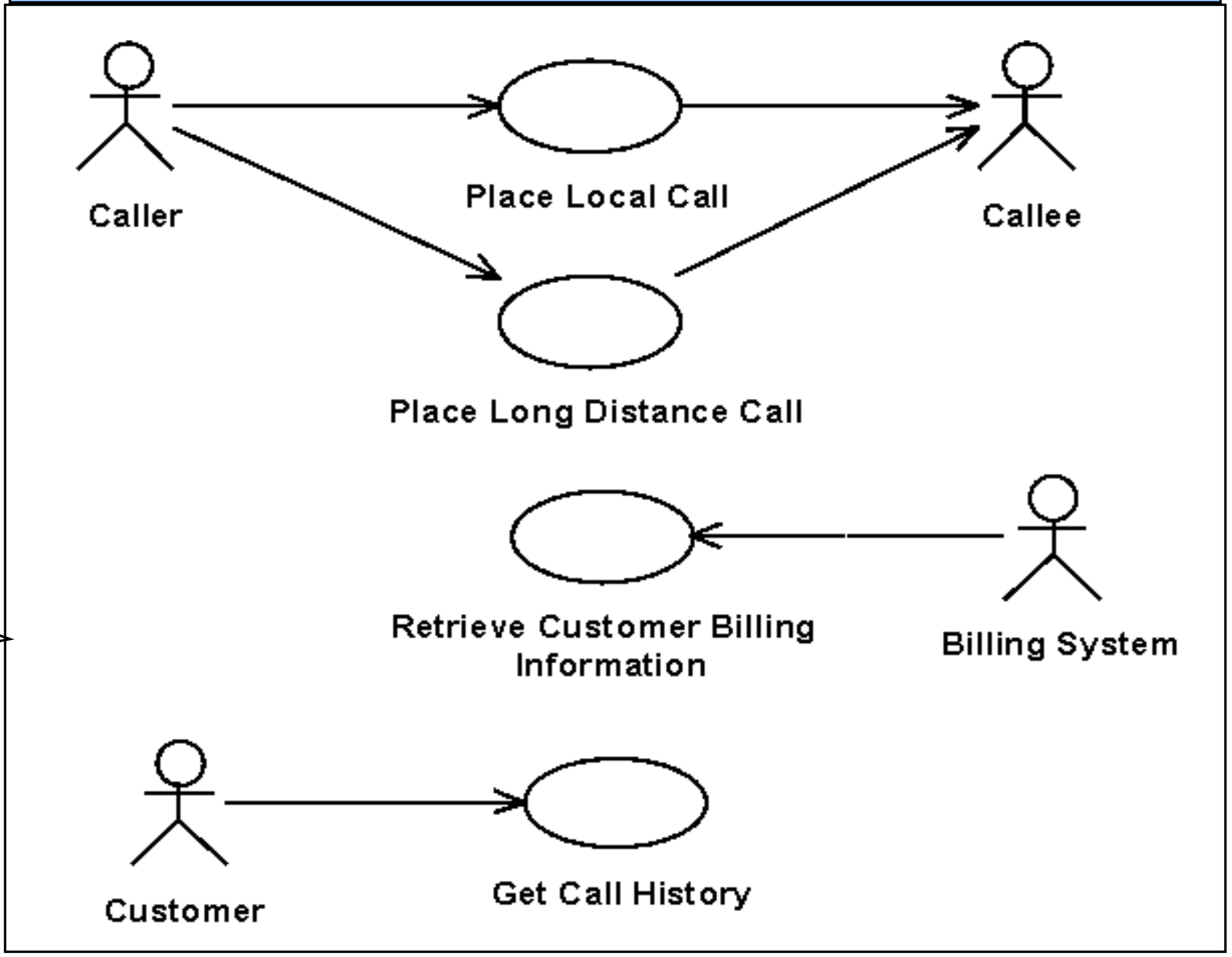
A use case:
- Is a sequence of actions a system performs that yields an observable result of value to a particular actor
- Is a collaboration between the system and its actors to deliver something of value for at least one of the actors.
- Is the smallest unit of activity that provides a meaningful resul user.
- Is self-contained, and alway the system in a consistent s

Essential Contents:
- 1..N **Use-Case Slices**

Described By:
- 1 **Use-Case Narrative**
- 1..N **Use-Case Realization**
- 1..N **Test Cases**

A set of use cases visualizes the scope and goals of a system in an easily accessible form.

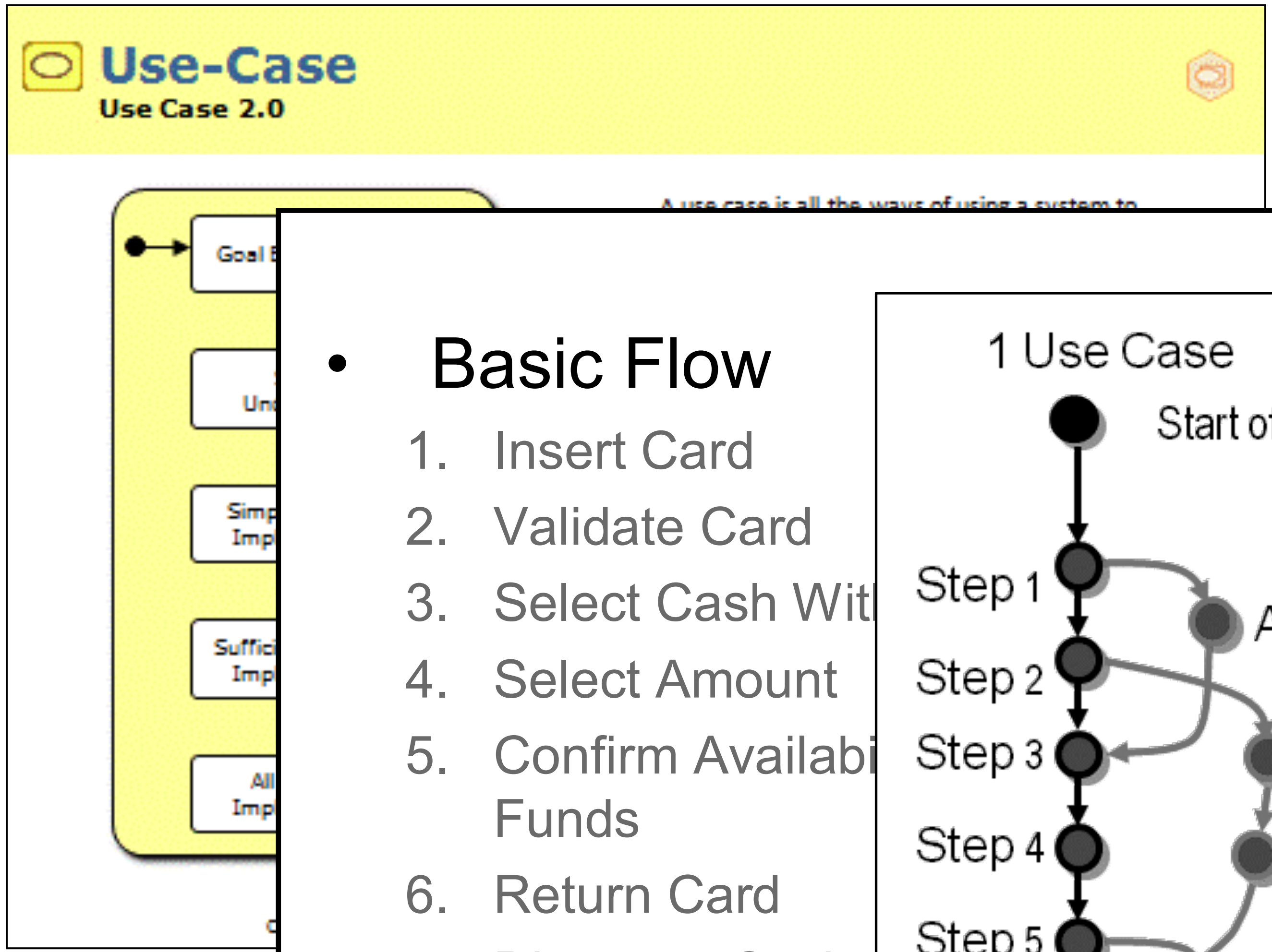
A Simple Telephone System



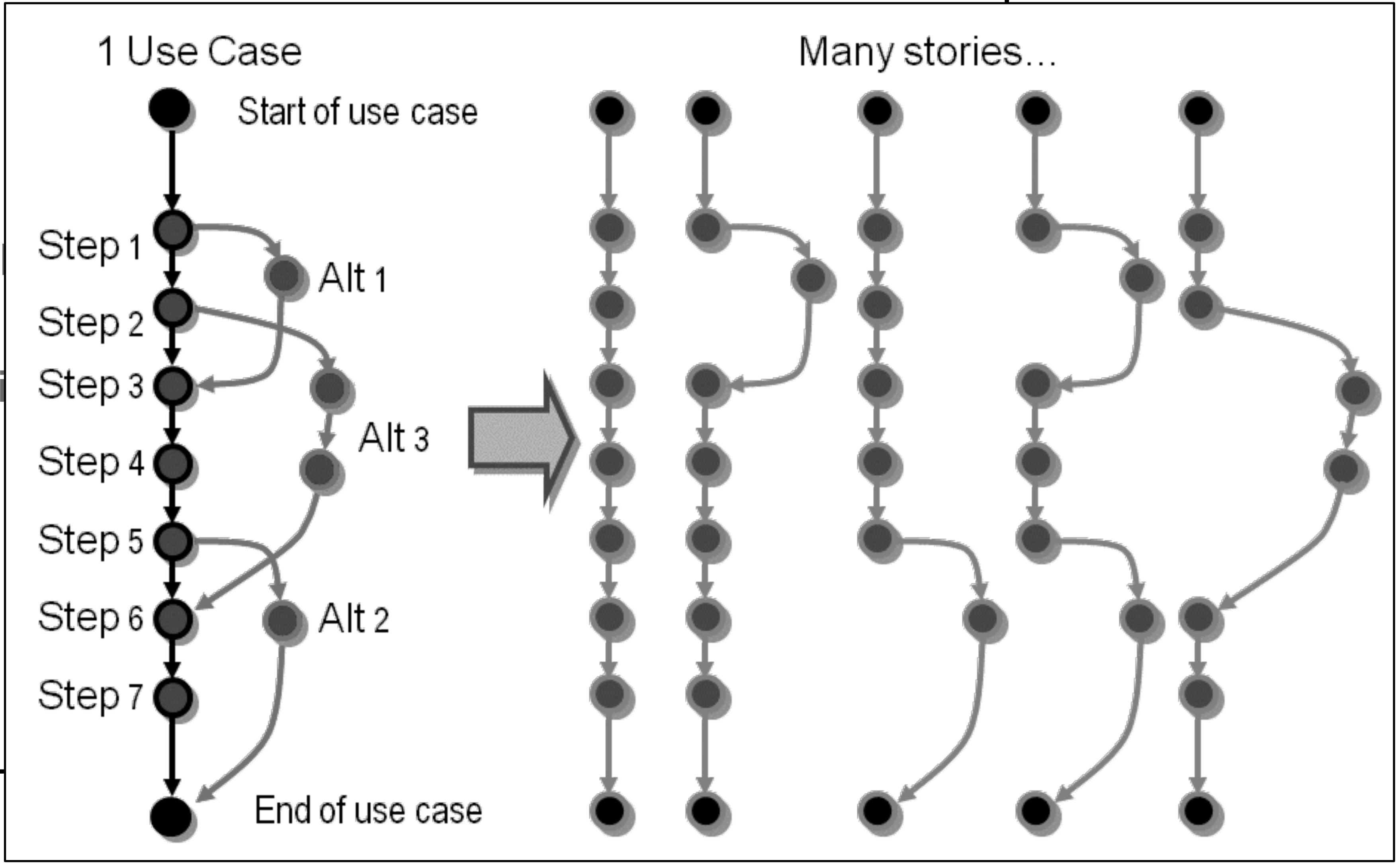
The use cases provide context for our conversations.

The use cases are our epics and themes.

A use case is represented by many stories….

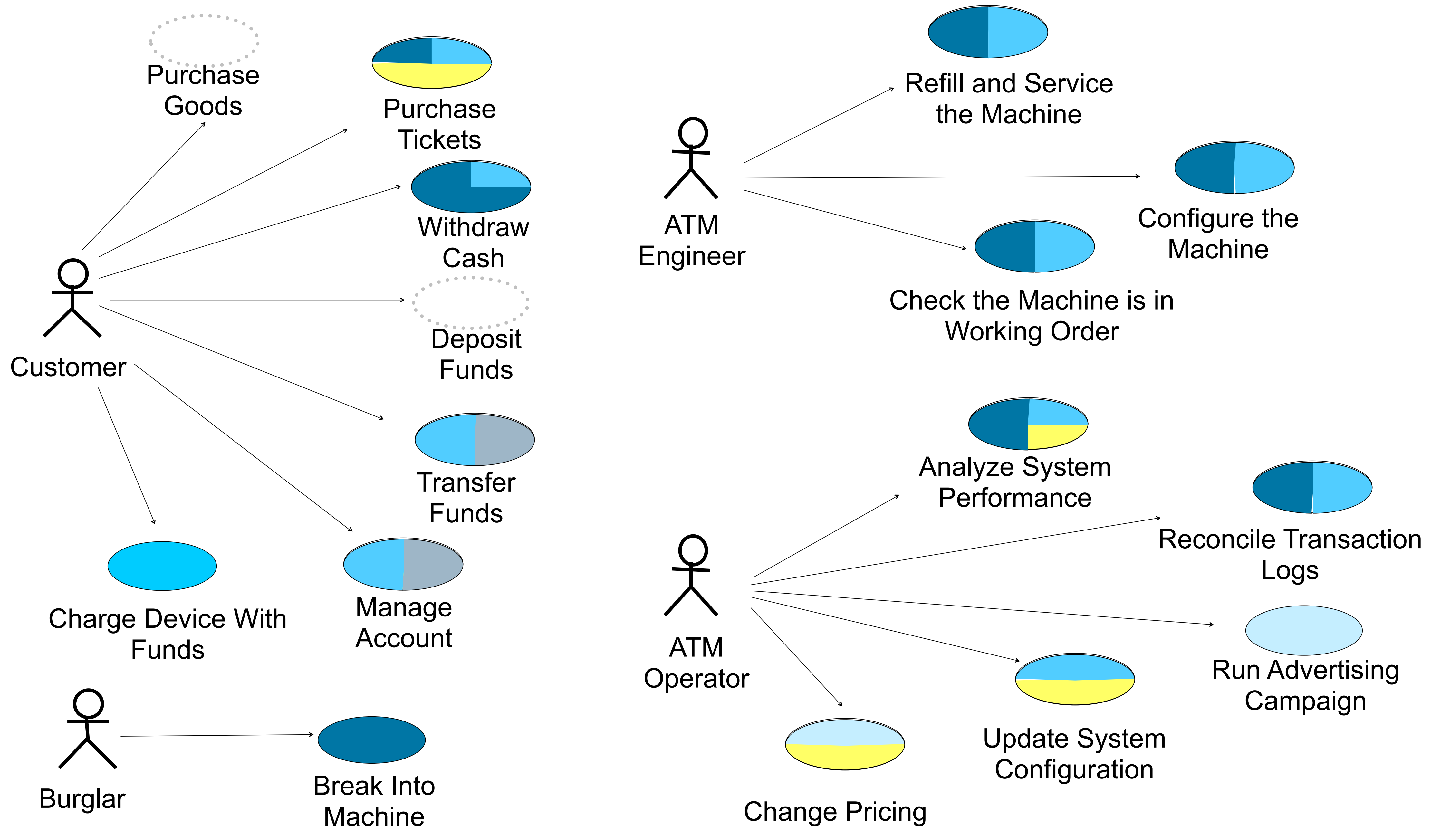


- Basic Flow
  1. Insert Card
  2. Validate Card
  3. Select Cash Wit
  4. Select Amount
  5. Confirm Availabi Funds
  6. Return Card
  7. Dispense Cash



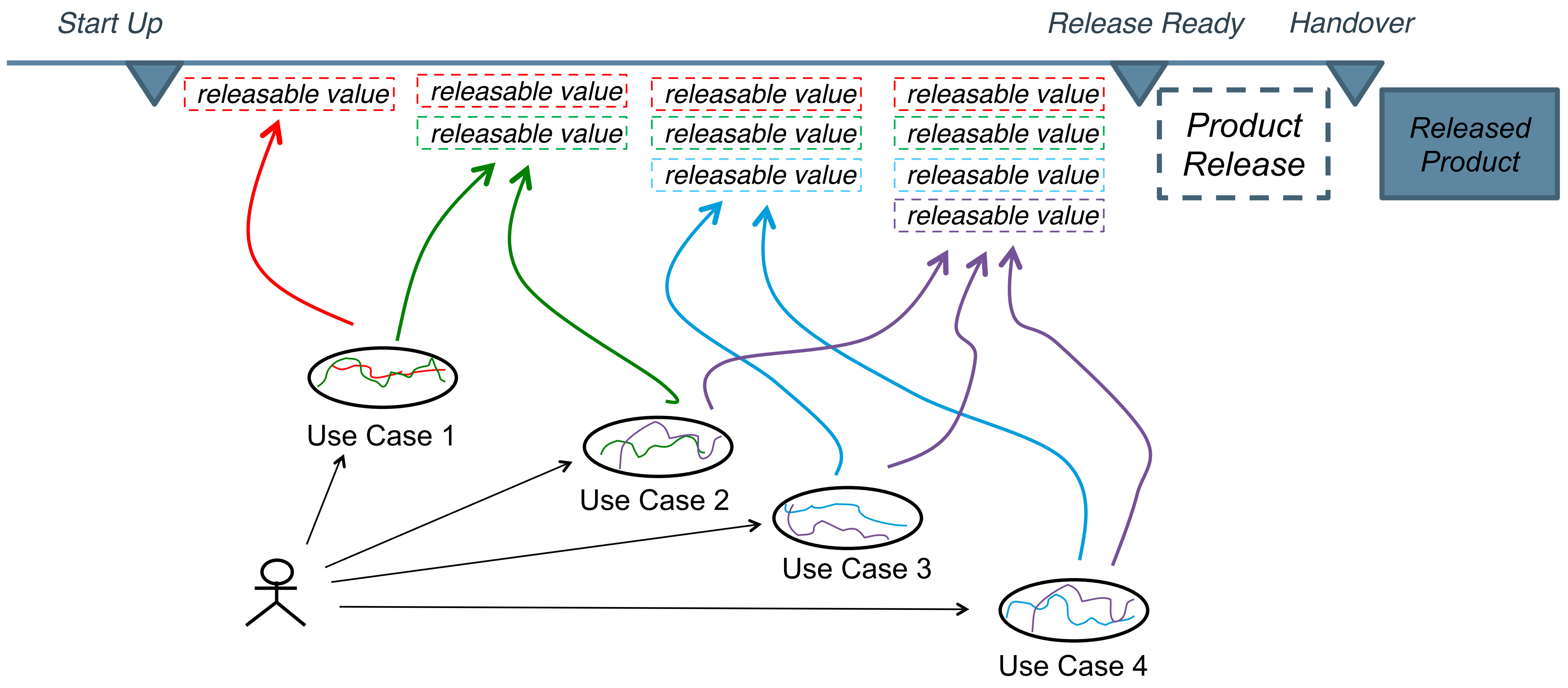
… often too many to code and test in one go.

Too many stories for a single release



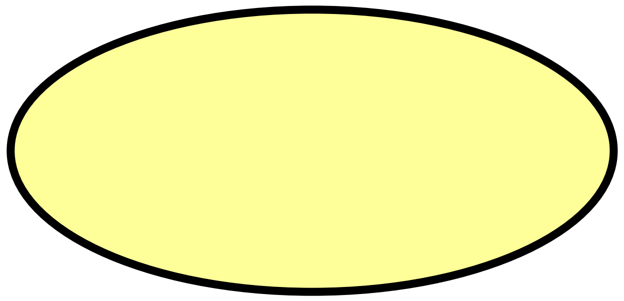
Release 1: ● , Release 2: ● ,
Release 3: ● , Release 4: ● , Out of Scope: ◌

Too many stories for a single increment



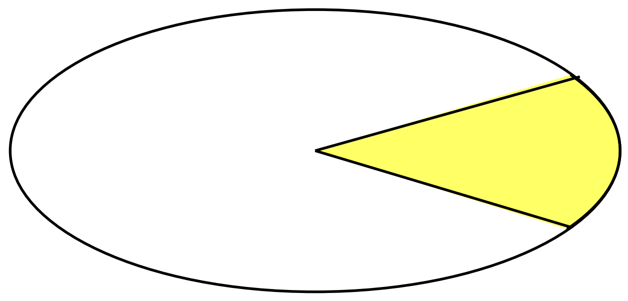
You need to slice up the use cases to provide stories suitable for iterative development, Kanban or Scrum.

So we slice up the use cases to drive the development



A Use Case

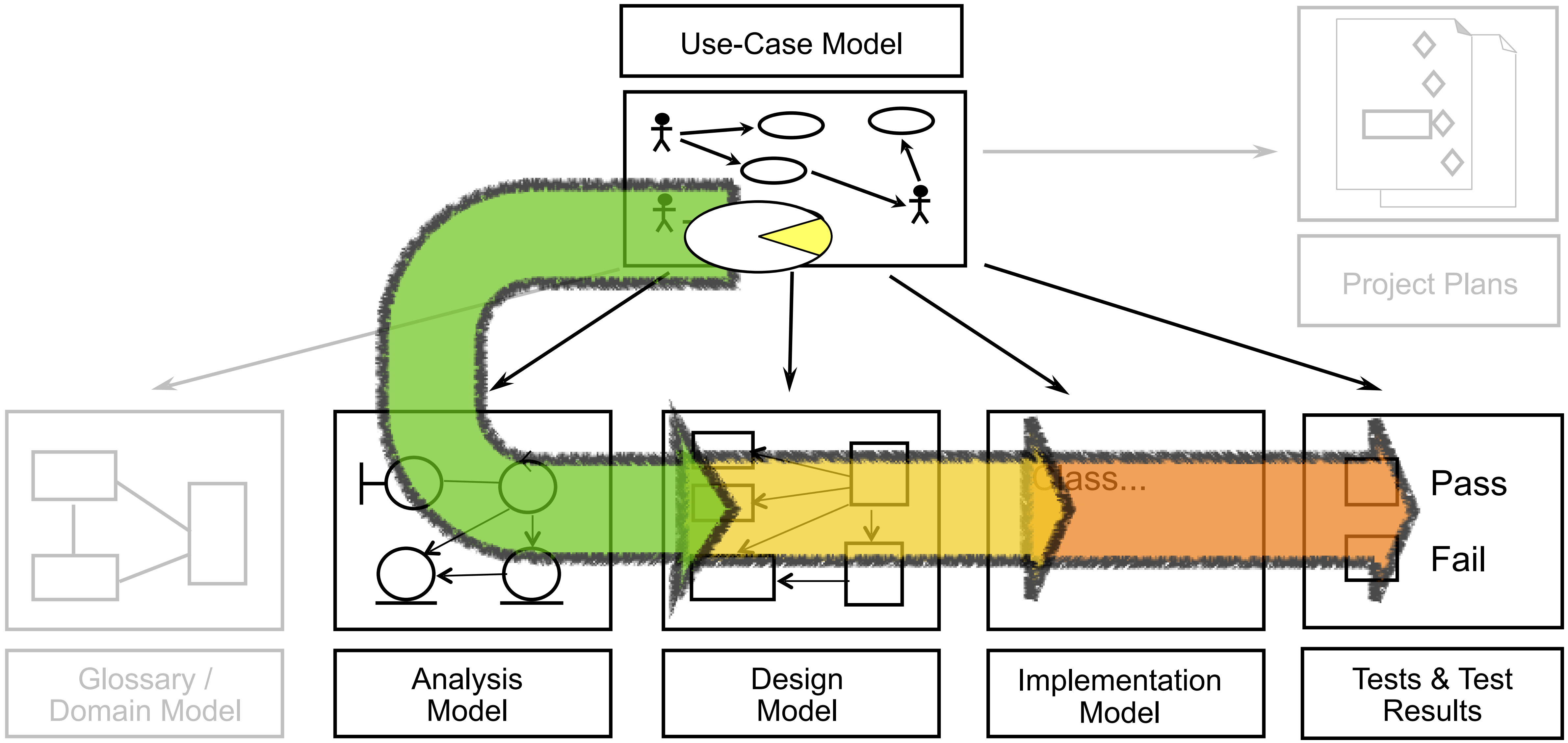
- Is described by a set of structured stories in the form of:
  - A use-case narrative containing flows and special requirements
  - And a set of matching Test Cases



A Use-Case Slice

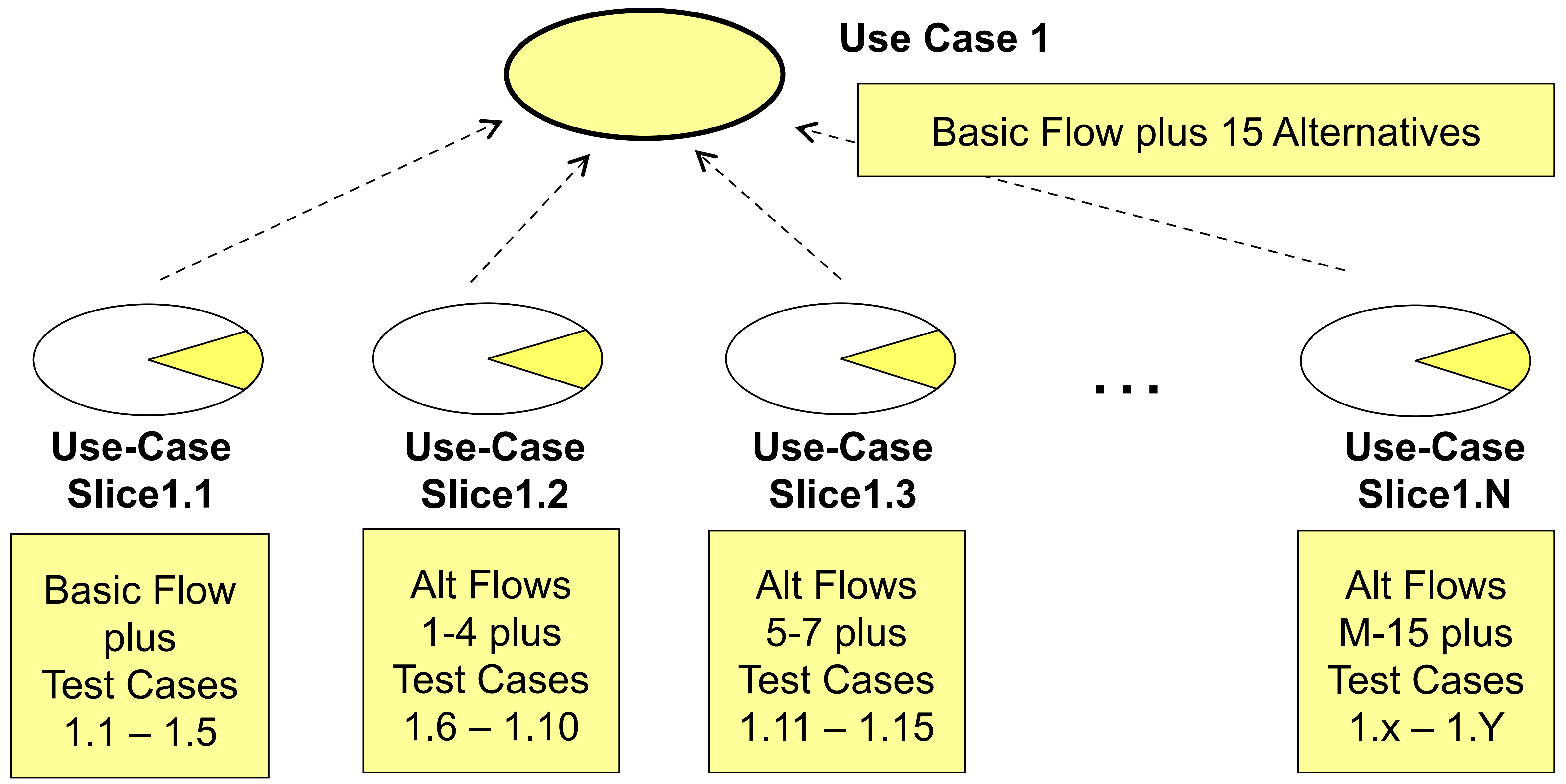
- Is created by selecting one or more stories for implementation
- …, acts as a placeholder for all the work required to complete the implementation of the stories
- …, and evolves to include the equivalent slices through design, implementation and test.

Use-Cases drive the development



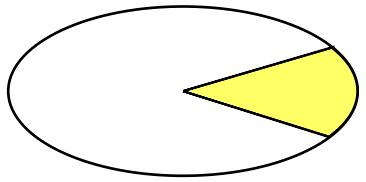
Use case slices are more than just the stories

A use case can usually be carved into several slices


Use Case 1
Basic Flow plus 15 Alternatives
. . .
Use-Case Slice1.1
Basic Flow plus Test Cases 1.1 – 1.5
Use-Case Slice1.2
Alt Flows 1-4 plus Test Cases 1.6 – 1.10
Use-Case Slice1.3
Alt Flows 5-7 plus Test Cases 1.11 – 1.15
Use-Case Slice1.N
Alt Flows M-15 plus Test Cases 1.x – 1.Y

Each slice is independently deliverable

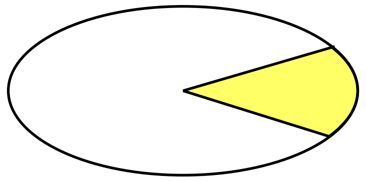
… and other quality attributes



Use-Case Slice 2.1

Basic Flow – Scenario 1 plus Test Case 2.1

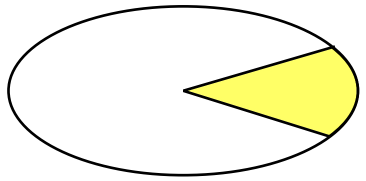
Slice1 – Build the basic flow and Test with one key scenario.



Use-Case Slice 2.2

Basic Flow – Rest of Scenarios plus Test Cases

Slice 2 – Complete the implementation and testing of the basic flow.



Use-Case Slice 2.3

Basic Flow – + Supp Req’t A, B & C Test Cases

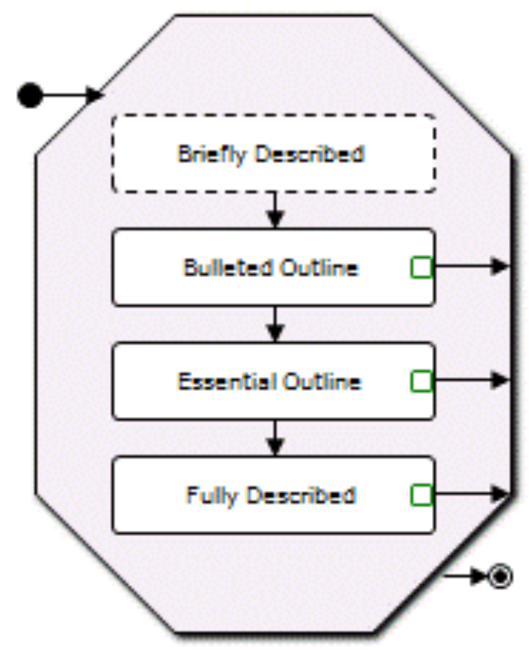
Slice 3 – Use the basic flow to performance and stress test the system.

You don’t even need the whole use case before you start slicing

IVAR JACOBSON INTERNATIONAL

You only need to outline the flows to create the slices

Use-Case Narrative
Use Case 2.0



Use-Case 1: Withdraw Cash

- 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

1.3 Card Handling Errors

Story: Wrong type of card.
Flows: BF, A1
Test Conditions

Story: Card St
Flows: BF, A7
Test Conditions
Points:
Default Amoun

1.2 Full Cash Withdrawal

Story: Empty an Account
Flows: BF
Test Conditions: Card 001, Account 001 containing £200
002

1.1 Normal Cash Withdrawal

Story: Withdraw Beer Money
Flows: BF
Test Conditions: Default Card, Default Account, Default Amounts

Points:

Use case narratives can be described at different levels of detail.
They start very lightweight by outlining the flows.

This is enough detail to identify our stories and define our slices.

IVAR JACOBSON INTERNATIONAL

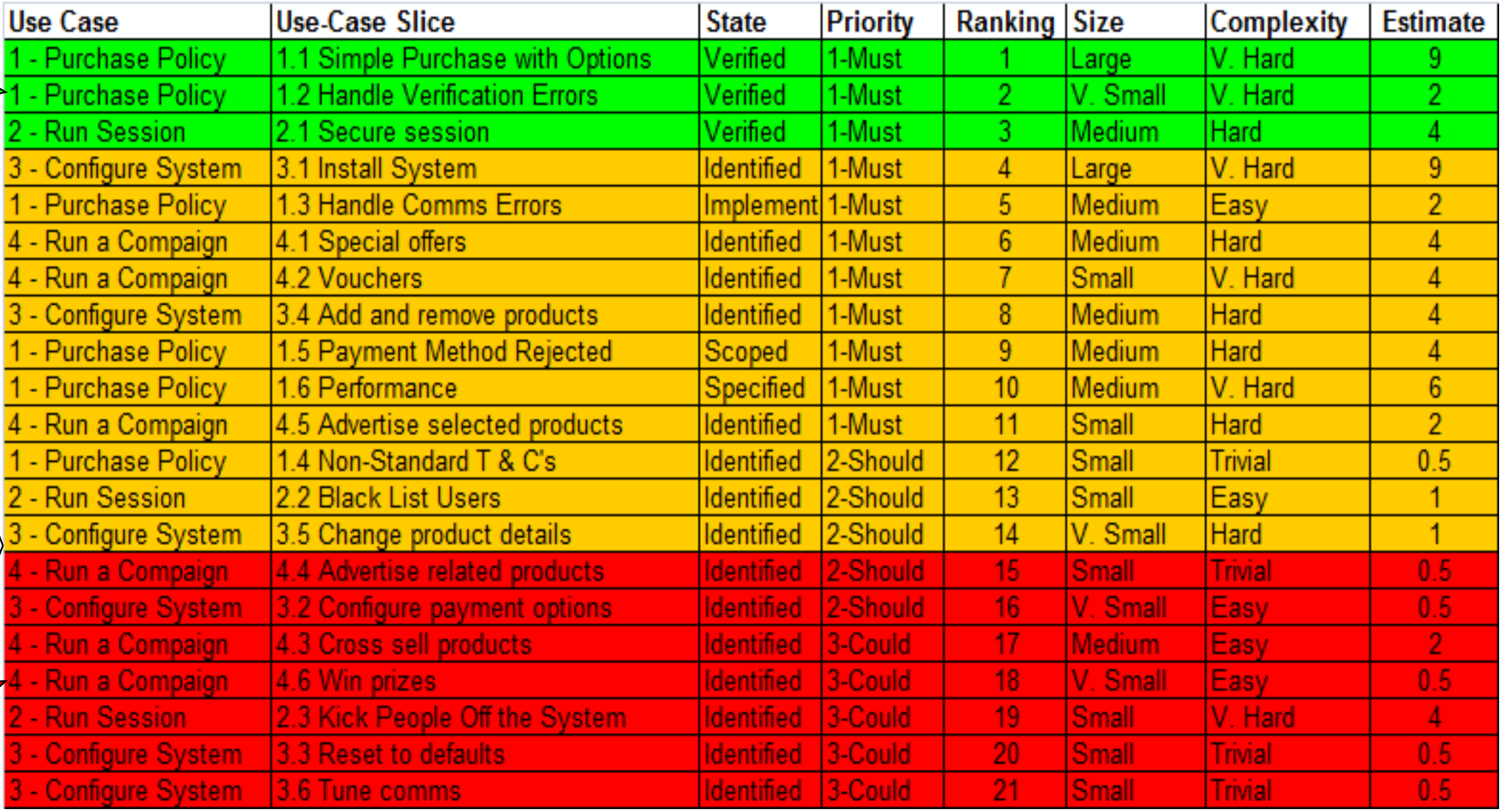
Slicing use cases to for product owners and development teams

Done

Doable

Where time runs out.

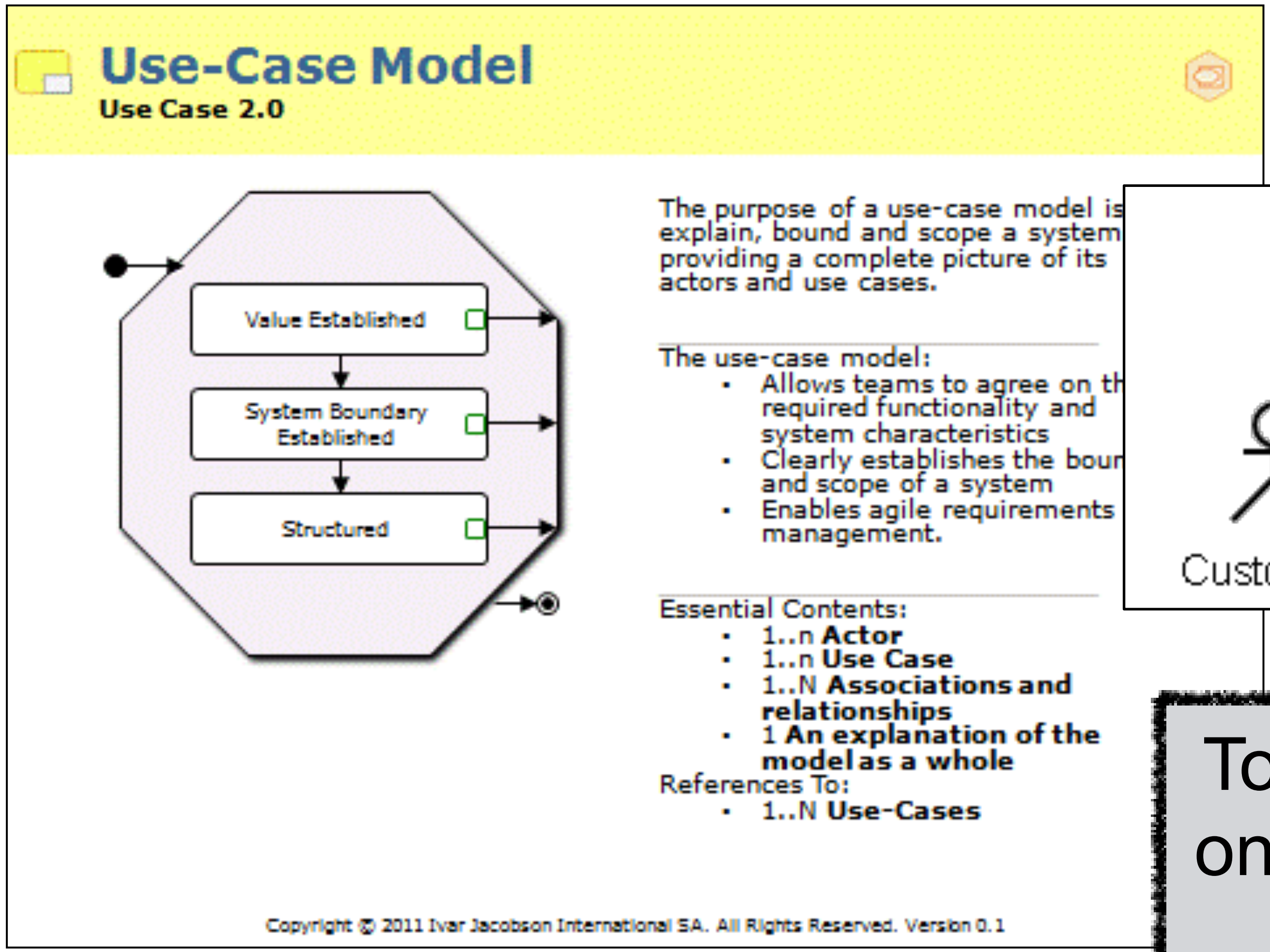
At risk



Use Case	Use-Case Slice	State	Priority	Ranking	Size	Complexity	Estimate
1 - Purchase Policy	1.1 Simple Purchase with Options	Verified	1-Must	1	Large	V. Hard	9
1 - Purchase Policy	1.2 Handle Verification Errors	Verified	1-Must	2	V. Small	V. Hard	2
2 - Run Session	2.1 Secure session	Verified	1-Must	3	Medium	Hard	4
3 - Configure System	3.1 Install System	Identified	1-Must	4	Large	V. Hard	9
1 - Purchase Policy	1.3 Handle Comms Errors	Implement	1-Must	5	Medium	Easy	2
4 - Run a Compaign	4.1 Special offers	Identified	1-Must	6	Medium	Hard	4
4 - Run a Compaign	4.2 Vouchers	Identified	1-Must	7	Small	V. Hard	4
3 - Configure System	3.4 Add and remove products	Identified	1-Must	8	Medium	Hard	4
1 - Purchase Policy	1.5 Payment Method Rejected	Scoped	1-Must	9	Medium	Hard	4
1 - Purchase Policy	1.6 Performance	Specified	1-Must	10	Medium	V. Hard	6
4 - Run a Compaign	4.5 Advertise selected products	Identified	1-Must	11	Small	Hard	2
1 - Purchase Policy	1.4 Non-Standard T & C's	Identified	2-Should	12	Small	Trivial	0.5
2 - Run Session	2.2 Black List Users	Identified	2-Should	13	Small	Easy	1
3 - Configure System	3.5 Change product details	Identified	2-Should	14	V. Small	Hard	1
4 - Run a Compaign	4.4 Advertise related products	Identified	2-Should	15	Small	Trivial	0.5
3 - Configure System	3.2 Configure payment options	Identified	2-Should	16	V. Small	Easy	0.5
4 - Run a Compaign	4.3 Cross sell products	Identified	3-Could	17	Medium	Easy	2
4 - Run a Compaign	4.6 Win prizes	Identified	3-Could	18	V. Small	Easy	0.5
2 - Run Session	2.3 Kick People Off the System	Identified	3-Could	19	Small	V. Hard	4
3 - Configure System	3.3 Reset to defaults	Identified	3-Could	20	Small	Trivial	0.5
3 - Configure System	3.6 Tune comms	Identified	3-Could	21	Small	Trivial	0.5

Building a backlog, tracking done, and knowing how much more you can do.

You only need to model what is important to you



Use-Case Model

Use Case 2.0

Value Established
System Boundary Established
Structured

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

The use-case model:

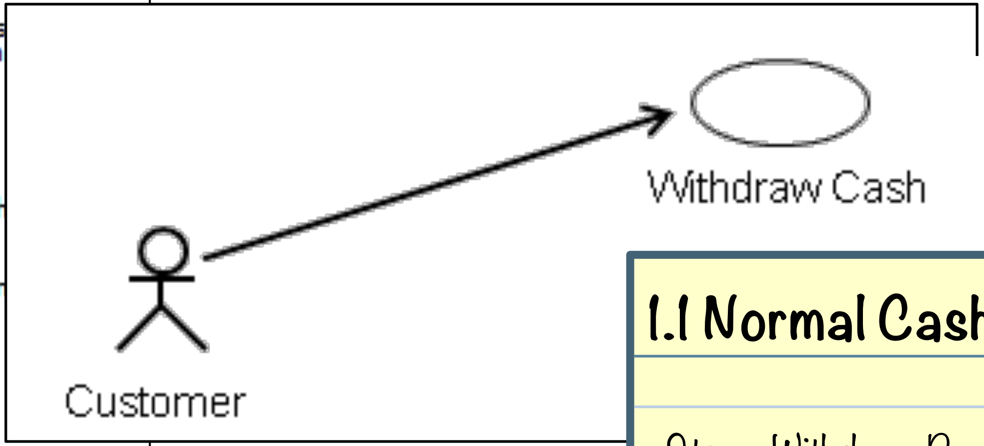
- Allows teams to agree on th required functionality and system characteristics
- Clearly establishes the boun and scope of a system
- Enables agile requirements management.

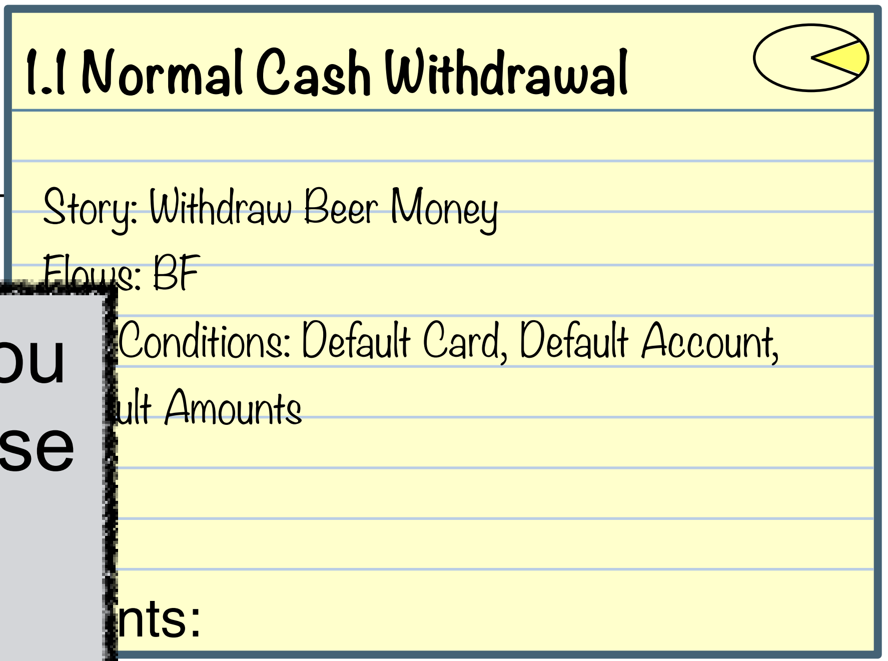
Essential Contents:

- 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-Cases**





1.1 Normal Cash Withdrawal

Story: Withdraw Beer Money

Flows: BF

Conditions: Default Card, Default Account,

ult Amounts

nts:

To get started you only need one use case and one slice.

The use-case model provides the big picture needed for effective scope management and release planning.

Agenda

- A Brief History of Use Cases
- Use-Case 2.0 – What is new?
- Use-Case 2.0 in practice
- Wrap Up

The agile sweet spot: slices with a small agile team

Building a web-based insurance application

- Small, co-located project team
- On-site product owner
- 4 then 2 week iterations
- Previous experience of use cases
- No experience of iterating

Lightweight use-case narratives to identify use-case slices.

The team wrote their test cases up front as they prepared their use case slices.

First working software within four weeks.

Use-Case 2.0 with very large-systems

Building a new banking straight-through processing engine

- Large distributed project team
- Many stakeholders and sponsors
- 6 then 4 and now 2 week iterations
- New to agile and iteration

Started with more formal use-case narratives and longer iterations.

Became more agile as they grew in confidence.

Delivered on-time and on-budget.

IVAR JACOBSON INTERNATIONAL

Working with external suppliers

- Innovation for Telecoms operations
- Working iteratively and incrementally through-out
- Requirements and testing in Holland
- All software development and testing out-sourced (much to India)
- Contractually need a formal requirements specification
- Many distributed teams – difficult to have timely conversations

Used outlines and use-case slices to identify deliverable pieces of work.

Evolved the use-case slices to provide clean clear orders for each iteration.

Created test cases up front and use these to QA the releases delivered by the supplier.

Using Use Case 2.0 for business change

- Agile business programs working iteratively and incrementally
- Business product managers creating software intensive products
- Adding these new products to existing enterprise IT systems
- Business requires synchronized up dates to multiple applications to meet their business needs
- Apply use-case 2.0 to the business as well as the software

Used outlines and use-case slices to identify slices of business change.

Analyzed the business use-case slices to identify the applications to be changed and their use-case slices.

Created "business" test cases up front and used these to QA the integrated set of applications provided by the IT Department.

IVAR JACOBSON INTERNATIONAL

Use Case 2.0 with Scrum: A winning combination

1. Slice early

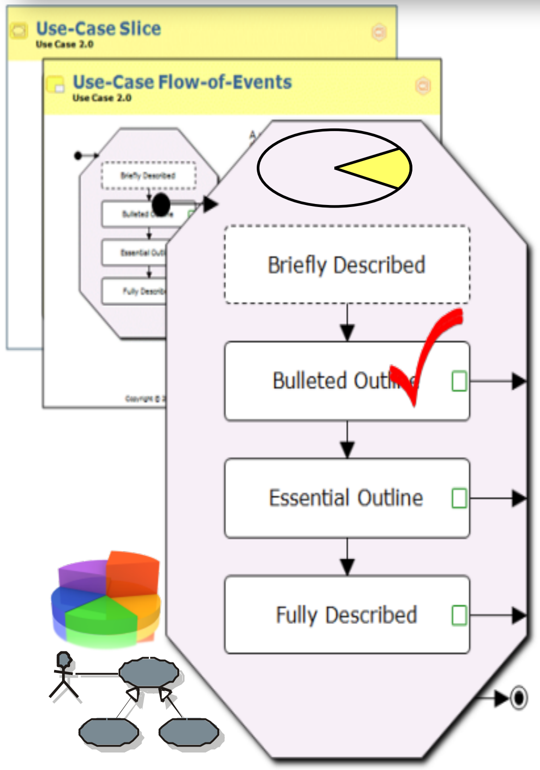
Use lightweight use case narratives to identify use-case slices and populate the backlog.

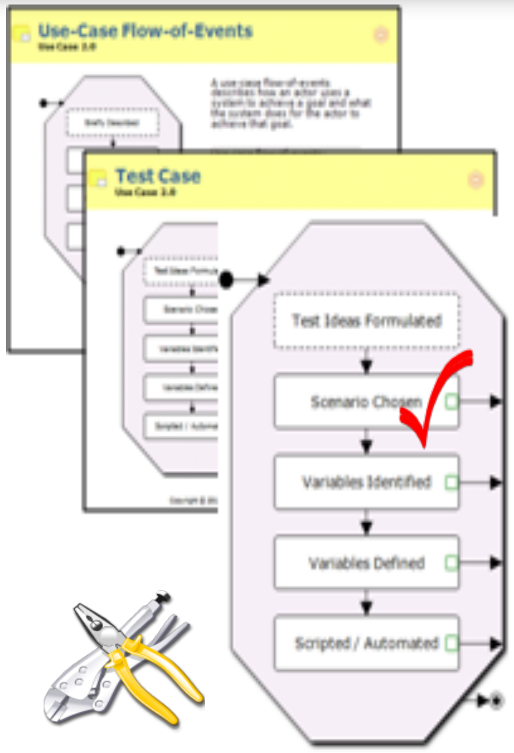
2. Test early and often

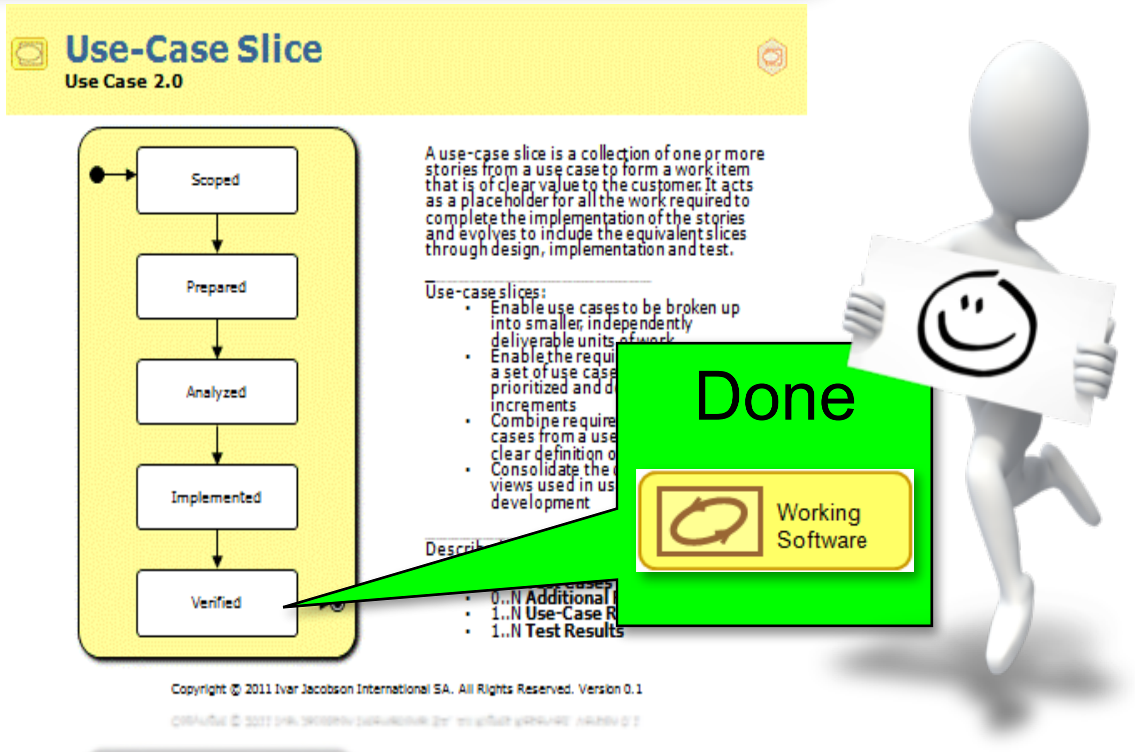
Write test cases up front to clearly define done, use these to demonstrate working software.

3. Deliver early and often

Use the use-case model to identfy the right slices to generate a usable system as early as possible.





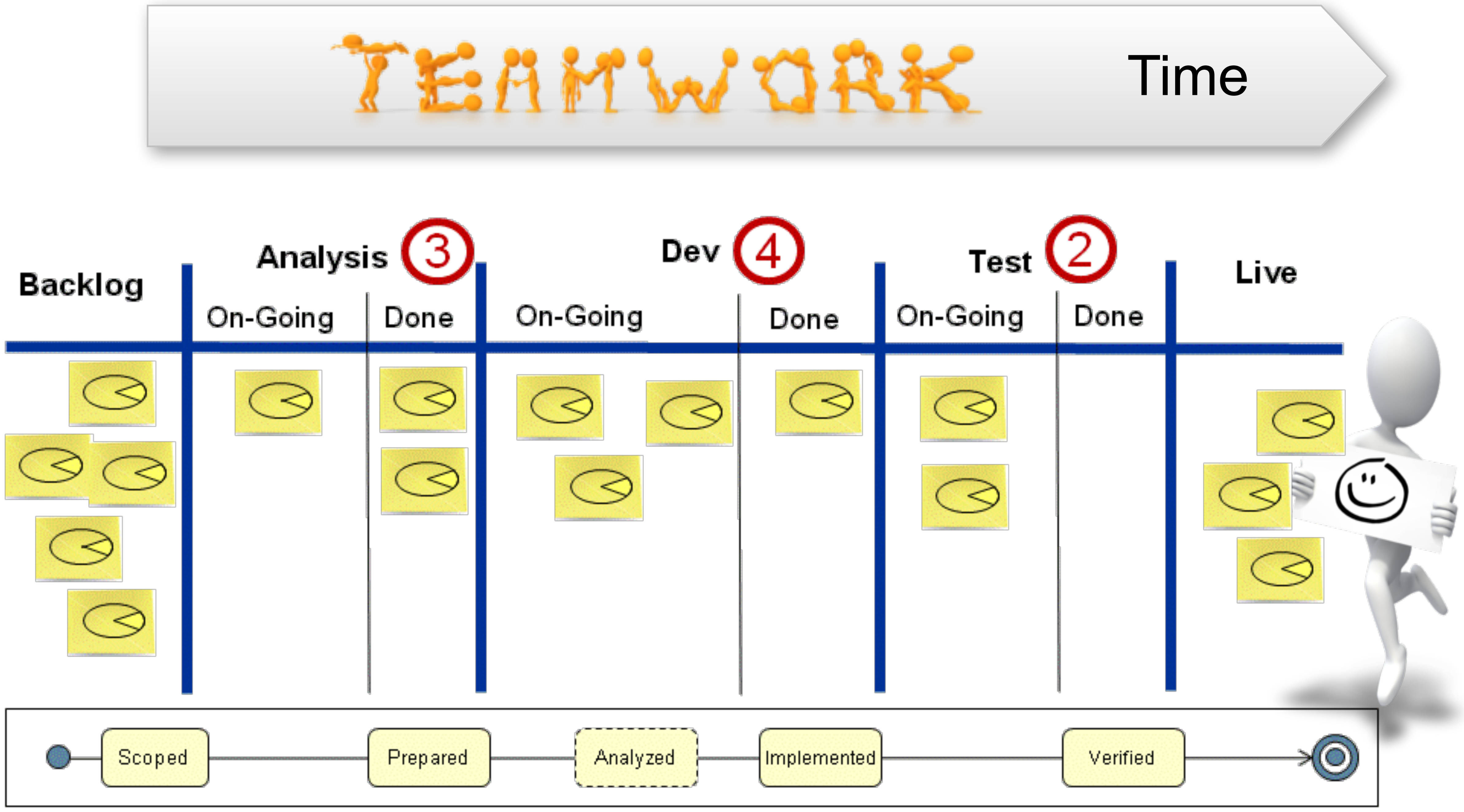


A use-case slice is a collection of one or more stories from a use case to form a work item that is of clear value to the customer. It acts as a placeholder for all the work required to complete the implementation of the stories and evolves to include the equivalent slices through design, implementation and test.

Use-case slices:
- Enable use cases to be broken up into smaller, independently deliverable units of work
- Enable the requi
- a set of use case
- prioritized and d
- increments
- Combine require
- cases from a use
- clear definition o
- Consolidate the
- views used in us
- development

Descri
- 0..N Additional
- 1..N Use-Case R
- 1..N Test Results

Use Case 2.0 with Kanban: A winning combination



Agenda

- A Brief History of Use Cases
- Use-Case 2.0 – What is new
- Use-Case 2.0 in practice
- Wrap Up

Summary - Use-Case 2.0

- Use cases are still use cases
- They provide context for our conversations
- We only model what is important
- We slice our use cases to drive the development
- We eliminate waste by using the lightest level of detail
- We include test cases (as part of the use case) to define done
- We use cards and backlogs to support agile ways-of-working
- We add detail to cope with out-sourcing and off-shoring
- We apply the techniques recursively to handle large projects, programs and business change

Use-Case 2.0

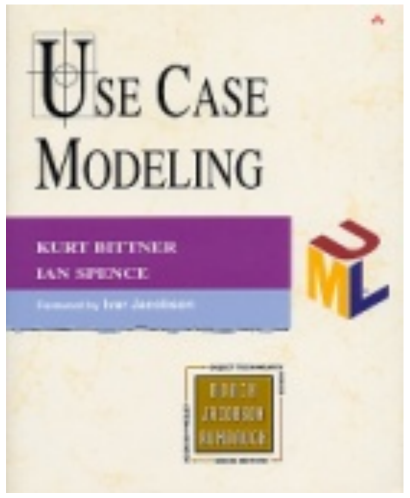
Agility in action.

The lightness of user stories with the power of modeling.

IVAR JACOBSON INTERNATIONAL

Use Case 2.0 -- Distinctive Features

- It helps you quickly understand the big picture
- As light as you want it to be
- Enabling incremental delivery
- It's not just about requirements, it's for the whole lifecycle
- It's also for non-functional requirements
- It's also for embedded software
- It's not just for software development – it's for business development as well
- Scaling to meet your needs – scaling in, scaling out and scaling up



Visit us at the IJI Stand to learn more and register for the free Use-Case 2.0 e-book

Questions

Thank You!

For questions, feel free to contact me, Kurt Bittner, at

kbittner@ivarjacobson.com

White papers and other resources can be downloaded from
www.ivarjacobson.com

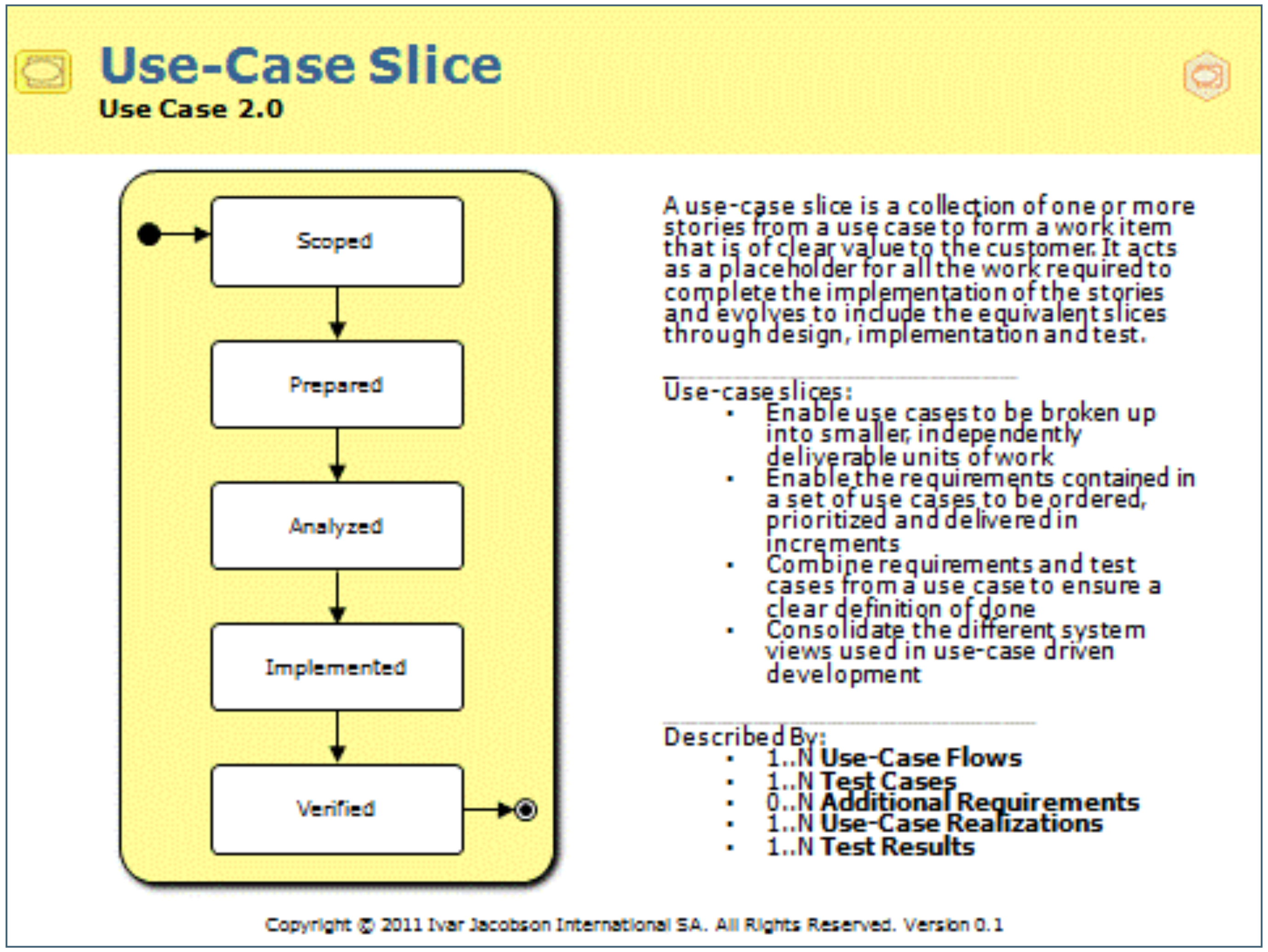
IVAR JACOBSON INTERNATIONAL

Use-Case 2.0
The lightness of user stories with the power of modeling

	User Stories	Use Cases	Added Value
Quick and Lightweight	Stories on cards	Bulleted outlines provide structured stories	Can be evolved to add detail where necessary
	Placeholders for conversations	Placeholders for conversations	Added context for the conversations
Work items for the backlog	Stories are small (1 to 5 ideal days to complete)	Slices are small (1 to 5 ideal days to complete)	Epics, themes, stories and user types brought together into one easily understood model
	Story cards can be ripped up and replaced if too large	Use-Case Slices can be ripped up and replaced if too large	Nothing is lost as we still have the model and the original use cases
Definitions of done	Confirmation via test cases added to card	Test cases are an integral part of the use case and the use-case slice	The use-case structure makes good test cases easy to find
	You never know when you've got all the stories	The model defines the whole system –easy to identify all the use cases and flows	The extent and scale of the system is readily apparent

IVAR JACOBSON INTERNATIONAL

You must include test cases to define when you are done

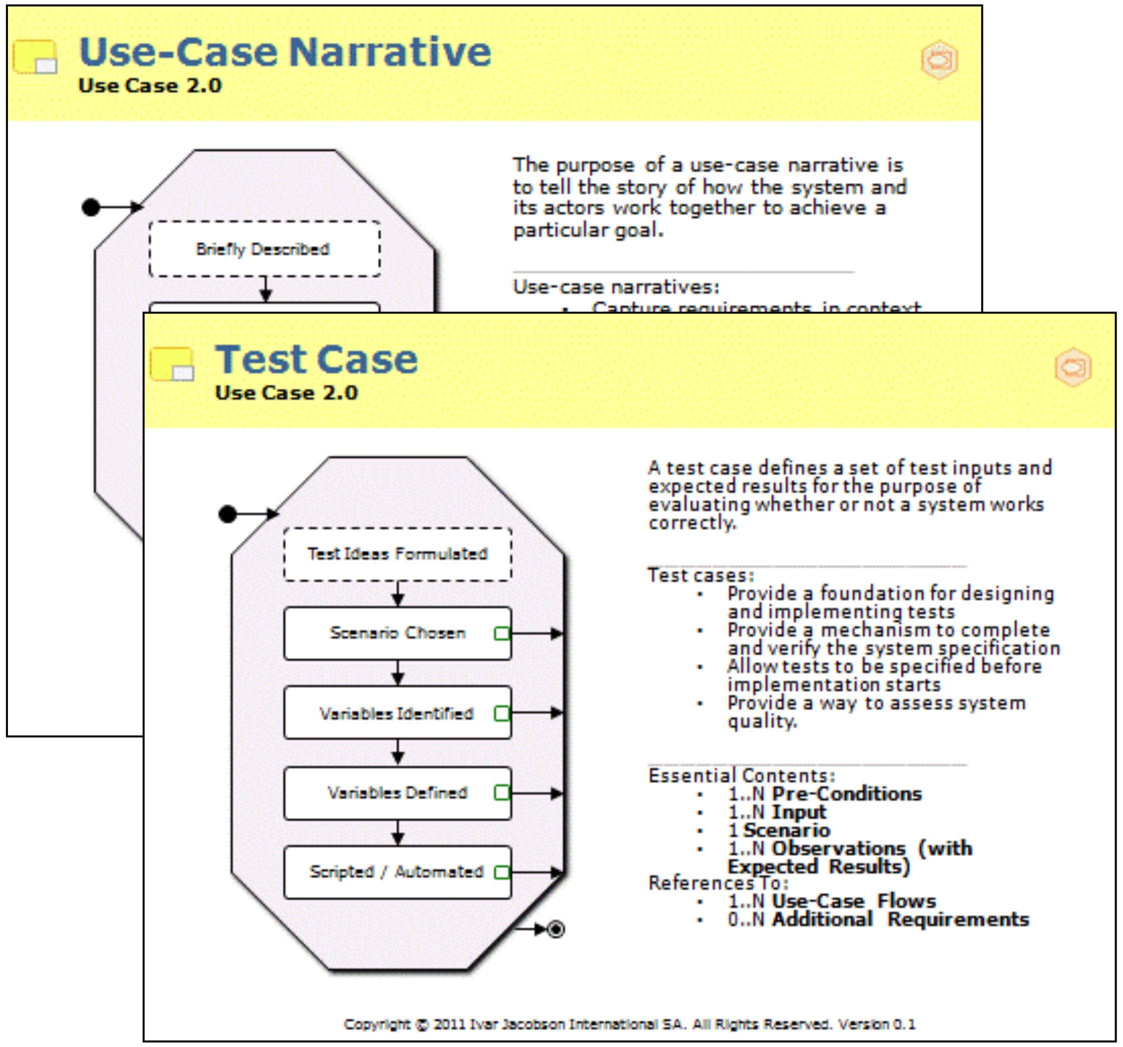

Scoped
Prepared
Analyzed
Implemented
Verified

Use-Case Slice
Use Case 2.0

A use-case slice is a collection of one or more stories from a use case to form a work item that is of clear value to the customer. It acts as a placeholder for all the work required to complete the implementation of the stories and evolves to include the equivalent slices through design, implementation and test.

Use-case slices:
- Enable use cases to be broken up into smaller, independently deliverable units of work
- Enable the requirements contained in a set of use cases to be ordered, prioritized and delivered in increments
- Combine requirements and test cases from a use case to ensure a clear definition of done
- Consolidate the different system views used in use-case driven development

Described By:
- 1..N **Use-Case Flows**
- 1..N **Test Cases**
- 0..N **Additional Requirements**
- 1..N **Use-Case Realizations**
- 1..N **Test Results**


Briefly Described
Test Ideas Formulated
Scenario Chosen
Variables Identified
Variables Defined
Scripted / Automated

Use-Case Narrative
Use Case 2.0

The purpose of a use-case narrative is to tell the story of how the system and its actors work together to achieve a particular goal.

Use-case narratives:
- Capture requirements in context

Test Case
Use Case 2.0

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

References To:
- 1..N **Use-Case Flows**
- 0..N **Additional Requirements**

Use-Case Slices -
bringing the Use-Case Narrative and the Test Cases together to define done.

IVAR JACOBSON INTERNATIONAL

Use-Case Narratives enable agility and scalability

Level of Detail	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 – provide the context for the conversations.	Scope management / use-case slicing Low fidelity estimation. Collaborative test definition Impact analysis and prototyping. Component identification Conversations in context
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.	Formal analysis and design Implementation and testing with full traceability Creation of user documentation. High fidelity estimation

IVAR JACOBSON INTERNATIONAL