BEDIFFERENT

ACE 2012 INTERNATIONAL

ACE 2012
INTERNATIONAL

Developing Deployment Use Cases

Building Use Cases as a component of effective requirements for an Aras Innovator project

Use Cases

- **A Use Case is:**
  - A story that describes the interactions between people and software… something that must happen in order for a user to accomplish something of value
- **A Use Case is NOT:**
  - A complete set of requirements for any project
  - A list of requirements, they do not describe business rules, data validations, technical interfaces, quality or platform specifications…..

Definitions

- **Use Case**
  - A story that describes the interaction between actors
  - A set of actions done by an actor to achieve a goal
  - Describes the use of a system to achieve a goal
  - One component of a full set of requirements
- **Requirement (IEEE Std 610.12-1990)**
  - A condition or capability needed by a stakeholder in order to solve a problem or achieve an objective
  - A condition or capability that must be met in order to satisfy a contract or regulation
  - The **documented** representation of a condition or capability

A Use Case

UC-1: Access ATM

Primary actor: Customer

Description: This use case describes the process that a customer would use to access an ATM main menu by swiping a valid ATM card

Pre-conditions:

Customer has a card

Customer has an account with a PIN.

ATM is active

ATM is ready for a new transaction

Success guarantees:

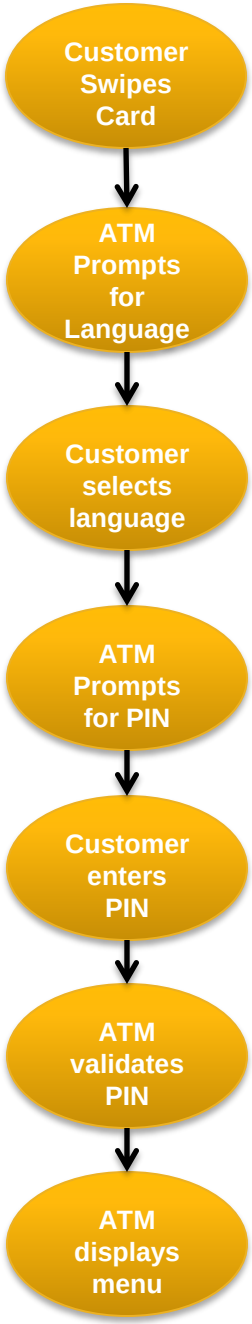
Customer is authenticated**.**

Menu of Services is displayed.

Trigger:

Main success scenario:

1. Customer swipes card.
2. ATM prompts for language.
3. Customer selects language.
5. ATM prompts for PIN.
6. Customer enters PIN.
7. ATM validates PIN.
8. ATM Displays Menu of Services.



Use Cases vs. User Stories

How are they different

User Story

- Written from the context of the user as a simple statement about their needs
- Generally written in the form of …

  Role needs to do something to achieve a benefit
- Sets the stage for a use case by stating the need before the use case describes the process
- User stories usually morph into business requirements and high level use cases

Anatomy of a Use Case

Name	The name of the use case in verb-noun format stated as the primary actor's goal *e.g. Create a customer purchase order*
Description	A short (1-2 Sentence) description of what the use case is intended to do or conditions for its use *e.g. This use case describes the actions required when a customer PO or Contract is received via email, fax or other means*
Level	The Level of detail at which the use case is written *Summary or user for our purposes*
Actors	Primary: The actor that initiates the Use Case Supporting: Any actor that interact with the use case
Pre-conditions	What must be true before the use case can begin. These are likely requirements as they will need to be validated before the use case starts
Success Guarantees	What interests are satisfied after successful completion of the use case
Trigger	The event that starts the use case
Main Success Scenario	The typical scenario in which the actor's goals are delivered. The happy path…. There is only one
Extensions	All other scenarios… Alternates and exceptions

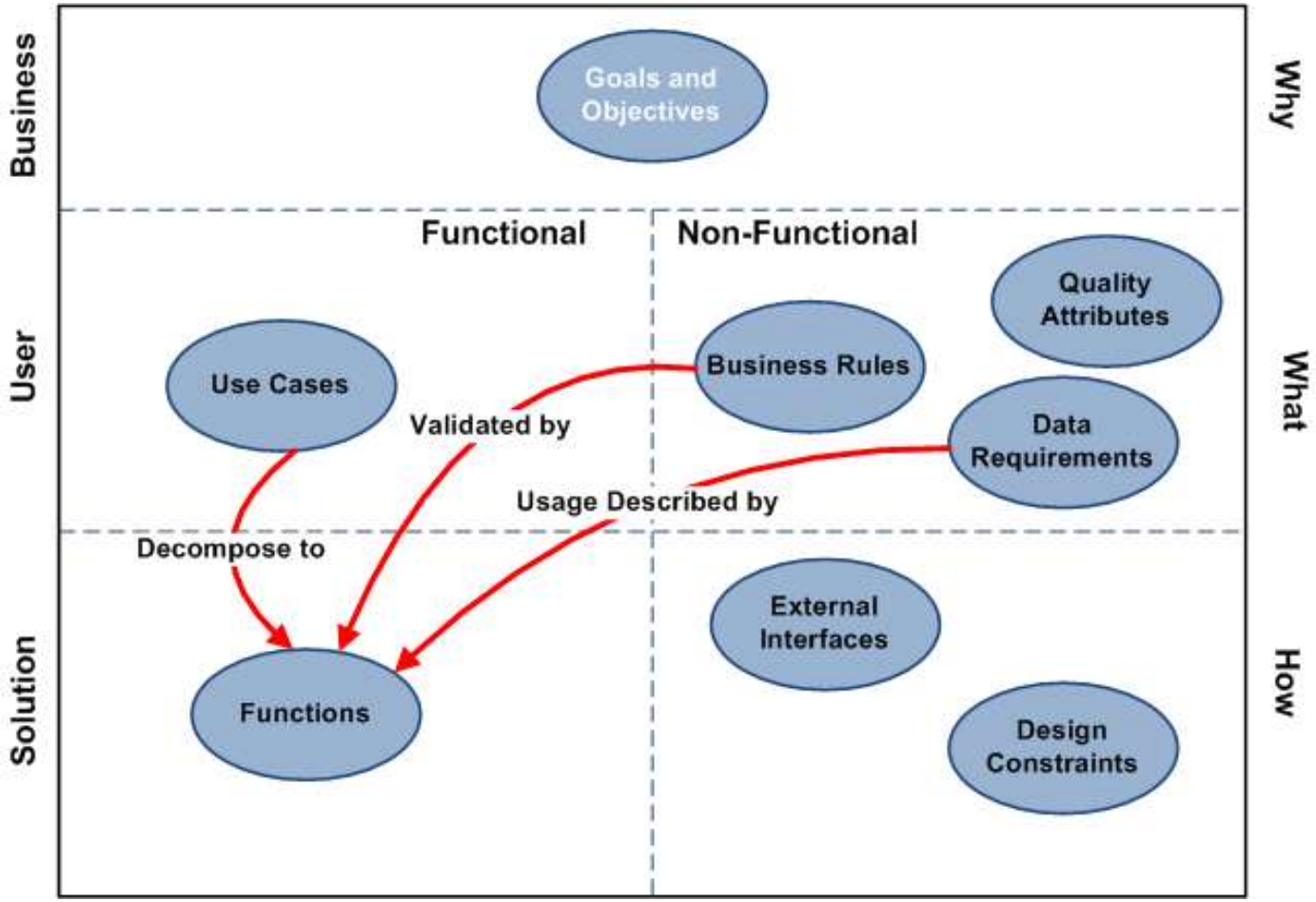
Use Case Standard Terms

- **System**
  - A process that accomplishes something of value in a business
  - A collection of interrelated items that work together
  - Can be a combination of people, hardware, software, etc
- **Actor**
  - Anyone or anything that interacts with a system
  - Can be:
    - A person by **role** or title – V.P of Marketing
    - Another system - MS Exchange
    - A functional area - Development

Requirements

- **Solution Requirements**
  - Specify parameters or components of the solution
    - Solution requires business process re-engineering
- **Functional Requirements**
  - Defines what the system is supposed to do
  - May be calculations, data manipulation, validations, etc
  - Generally expressed in the form "system must do……”
- **Non Functional Requirements**
  - Usually specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.
  - Generally expressed in the form "system shall be….."

Big Picture View



Functions must do something because of business or data requirements

Characteristics of Effective Requirements

Requirements MUST be:

- **Necessary**
  - Must support a business goal
- **Testable or verifiable**
  - Must be able to create a clear test and/or demonstrate functionality
- **Clear, Concise, & Singular**
  - Not ambiguous or vague
  - Leads all readers to the same interpretation
- **Consistent**
  - Not conflicting with any other requirement
- **Traceable**
  - Can be connected back to a source or forward to a use case
- **Documented**

Characteristics of Effective Requirements

Requirements Should be:

- **Feasible**
  - Possible to implement within the limitation of the technology
- **Negotiable**
  - Tradeoffs are made with stakeholders
- **Prioritized**
  - Could be as simple as “Must Have” and “Nice to Have”
- **Not dependent on design**
  - State ***what*** needs to be done not ***how*** is should be done

Use Cases & Requirements

- **Use Cases document a portion of a complete requirements set**
  - Sometimes only 1/3 or so of a complete set
- **Use Cases describe functional requirements**
  - Provide context to requirements
  - Provide a sequence of events
- **Traditional requirements don’t provide context**
  - Typically a list of unrelated requirements
  - Can be difficult to read and understand

Writing Use Cases

Example Use Case

UC-1: Access ATM

Primary actor: Customer

Description: This use case allows bank customers to log in to an ATM terminal in order to access remote banking services. This use case describes the process for an ATM in which the card is swiped (not taken).

Pre-conditions:

Customer has a card of the correct form factor to fit the card reader.

Customer has an **Account with a PIN.**

ATM is online and in service.

Bank Welcome Screen is displayed, with instruction to swipe card.

Success guarantees:

Customer is authenticated**.**

Menu of Services is displayed.

Trigger:

Main success scenario:

1. Customer swipes card.
2. ATM prompts for language.
3. Customer selects language.
4. ATM reads **card.**
5. ATM prompts for PIN.
6. Customer enters PIN.
7. ATM validates PIN.
8. ATM Displays Menu of Services.

Extensions:

***.a. At any time during the transaction, Customer selects "cancel" option:**

1. ATM displays message (e.g. "are you sure?").
2. Customer selects "yes" option.
3. ATM Displays Bank Welcome Screen.

4.a. ATM can't read card:

1. ATM displays message (e.g. "can't read card").
2. ATM displays prompt to re-swipe card.
3. Return to step 1.

7.a. Wrong PIN:

1. ATM displays message (e.g. "wrong PIN").
2. Return to step 5.

7.b. Wrong PIN 3 times:

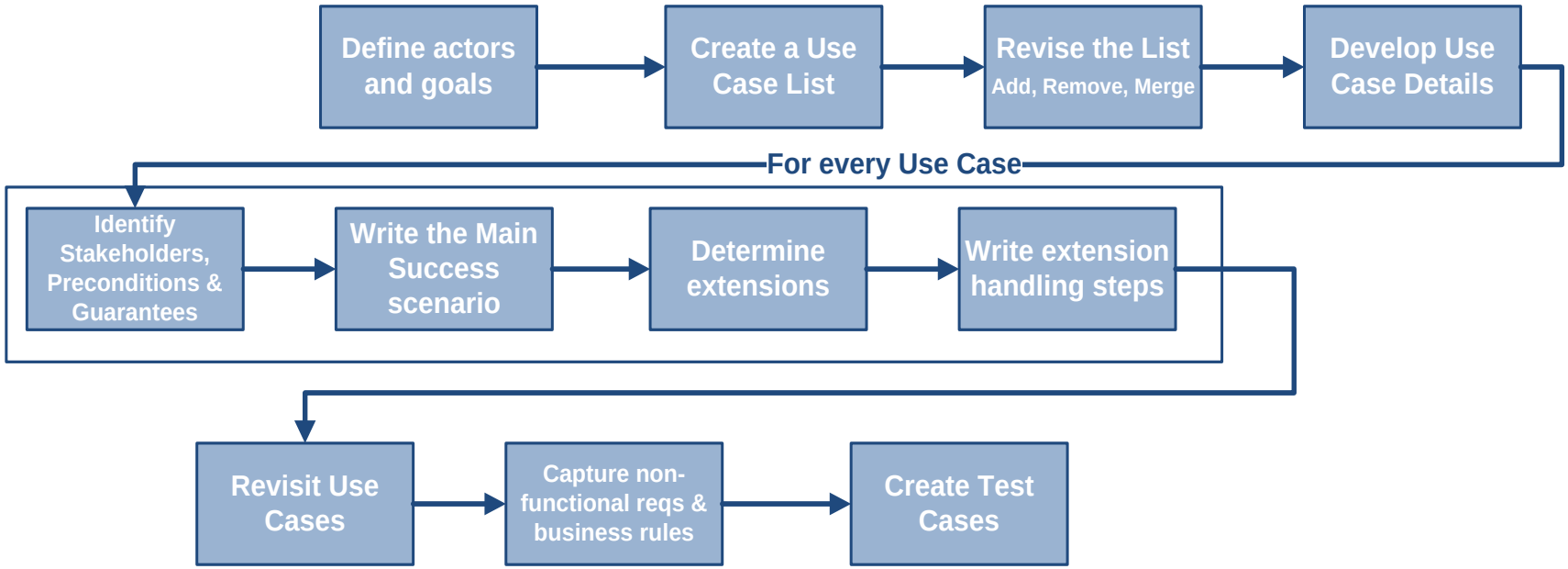
1. ATM displays message (e.g. "there's been a problem…").
2. ATM sends message to bank to lock account.
3. ATM Displays Bank Welcome Screen.

Still a lot Missing Here

What you don't see in this Use Case…
Business Rules

- **Language Support**
  - What languages are supported
- **Issues with card reading**
  - What happens if the card is unreadable
- **Credentials**
  - PIN entered must equal PIN Stored?
  - Number of tries?
  - Max number of tries?
- **Response time**
  - What is the max wait time for the system

A Use Case approach



Actors and Goals

- **Actors**
  - People (usually defined by role)
  - Functional areas
  - Other systems..
  - Primary Actor: the actor that initiates the process
- **Goals**
  - What an actor needs to accomplish with the system
  - Usually turn into first use cases
  - 2-3 words in verb-noun format - Create ECNs
  - Should be from primary actor's perspective
  - Focus on one actor at a time

Brainstorm actors and goals

Actor & Goal List

Actor	Goal
Sales	Create Quotes
	Publish Quotes
	Email Quotes
Finance	Process PO
	Process Invoice
	Email Customer

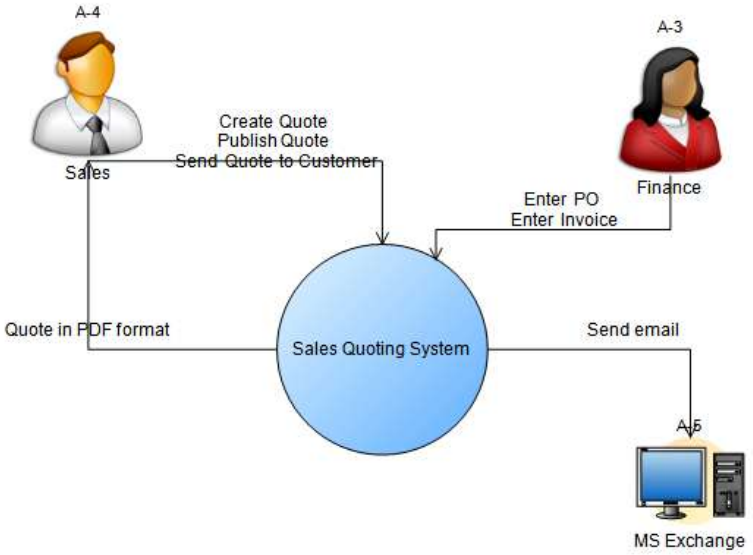
Sales
- Create Quotes
- Publish Quotes
- Email Quotes

Etc

Not used for anything more than brainstorming
Can be a simple MS Word indented list or table
Not really maintained after use cases are created

Context Diagram

A Brainstorming Tool



- High level view of the system
- Shows actor input, system output
- Does not show a sequence of events
- Does not show conditional events
- Does not show interactions between actors
- Useful for a quick overview of the solution
- A good brainstorming tool

Identifying Use Cases

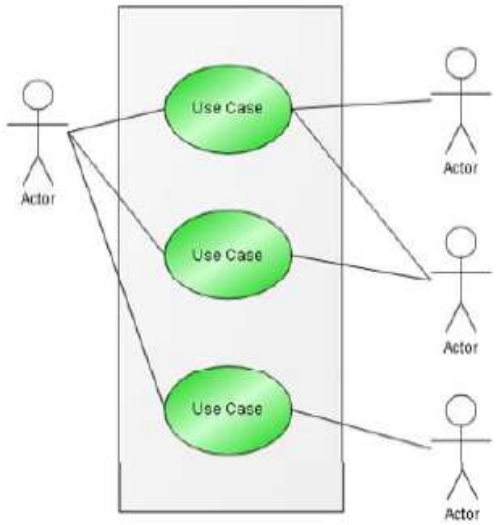
some approaches

- **Use a Context diagram if you have one**
  - For every input, there is a use case to do something with it
  - For every output, there is a use case to produce it
- **Use actor/users goals**
  - These goals usually turn into a high level use case
- **Identify events that must happen**
  - What causes an actor to interact with the system
  - When does the system need to produce output
  - What triggers the system to do something

Use Case List and Diagrams

- **It is what is says it is !!**
  - A list of high level use cases identified during brainstorming
- **Intended to show the use cases that will be addressed by the system**
- **Shows how actors interact with Use Cases**

Use Case Diagram



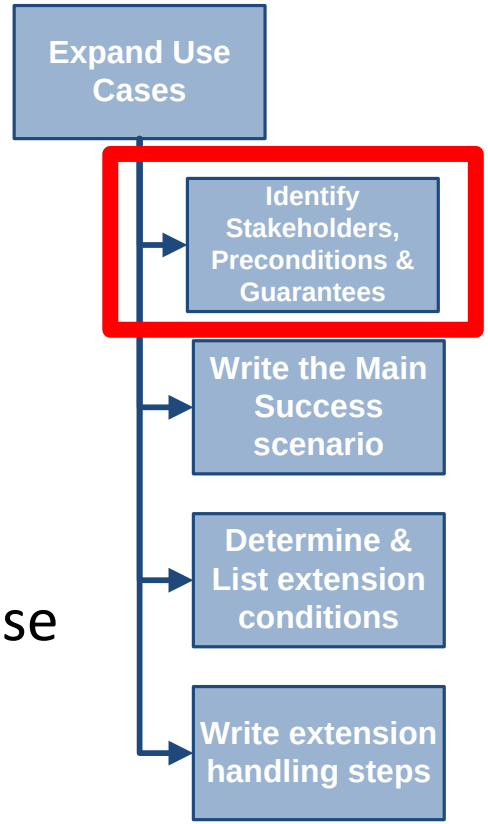
Use Case List

#	Name	Primary Actor	Scope	Priority	Complexity
1	Create an ECO	World	In	High	Low
2	Submit an ECO	World	In	High	Low
3	Approve an ECO	CCB	In	High	Low
4	Search for an ECO	World	In	Med	Low

Revising the Use Case List

- **Take a second look at the list**
  - Review the list of events
  - Review actor goals
- **Review the list as needed**
  - Add missing use cases
  - Merge smaller use cases where possible
  - Identify use cases that need to be broken into additional use cases
  - Remove any that are not in scope

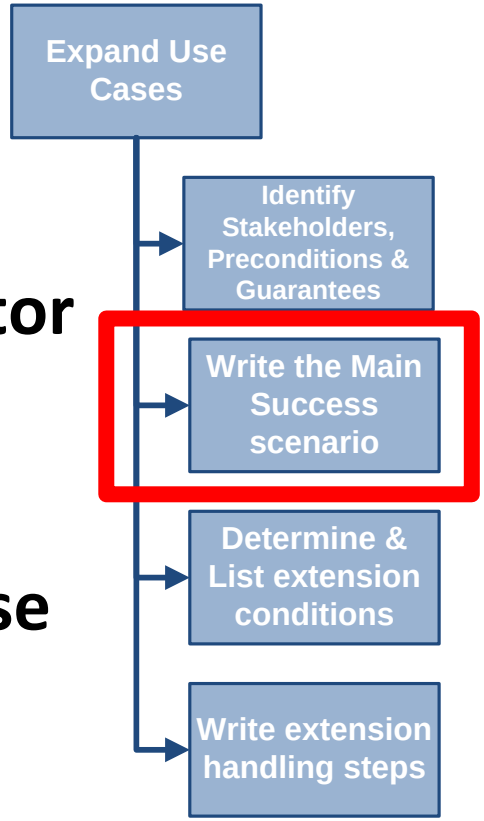
Expanding Use Cases
Stakeholders & Preconditions



- **Stakeholders**
  - All actors are stakeholders
  - All stakeholders do not need to be actors
- **Pre-conditions**
  - What must exist for this Use Case to begin
  - Could be the completion of another Use Case
- **Trigger**
  - An action by the primary actor that the system can detect
  - Can sometimes be specified as the first step in the Use Case

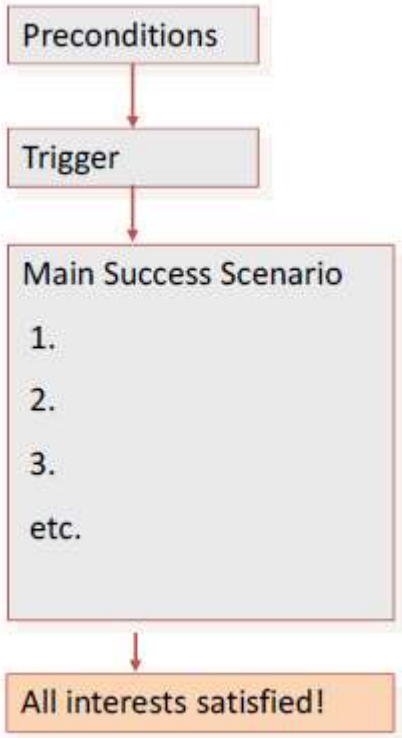
Expanding Use Cases
Main Success Scenario

- **Write the main success scenario first**
  - The Happy Path, perfect world, etc
- **Describe the interaction between the actor and the system from the trigger to the target**
- **There is only one happy path per Use Case**
- **Define the success condition and failure condition**



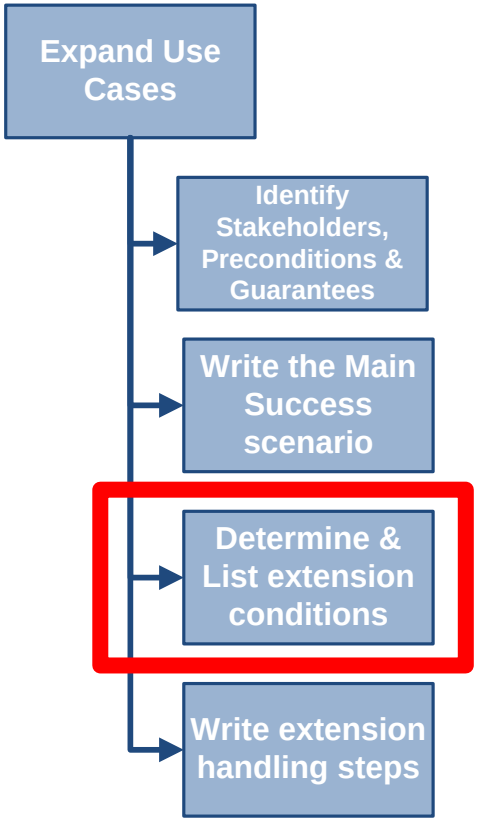
Use Case Flow

- List the conditions that MUST apply before starting
- List the event that starts the Use Case
- Describe the Happy Path
- List the success condition



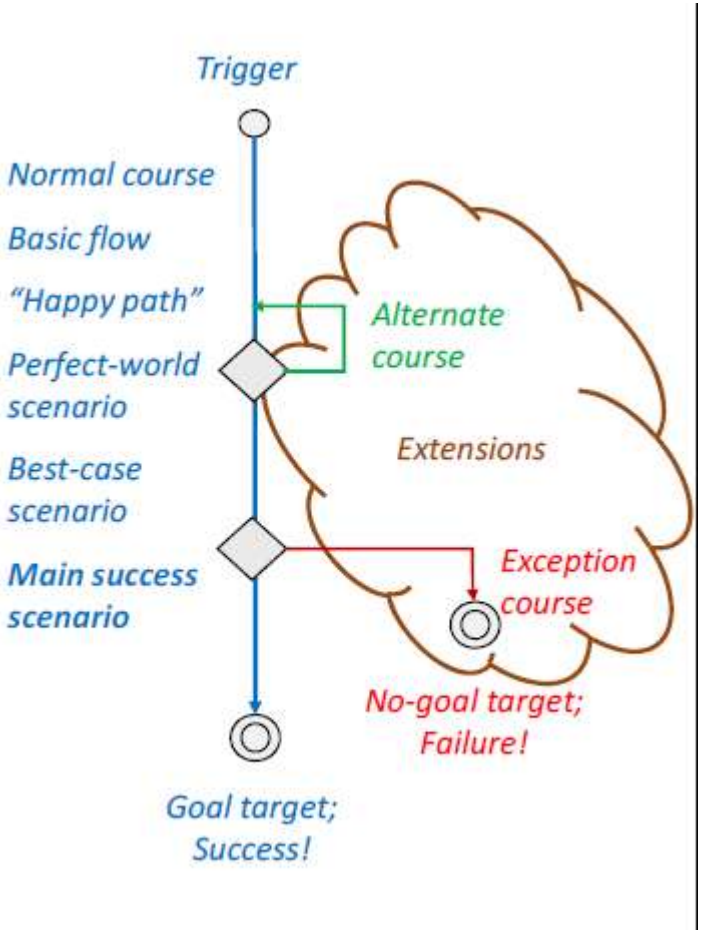
Expanding Use Cases
Determine the extensions

- **Scenarios that happen outside the happy path**
  - Result in different behavior
  - Could join back up with the happy path
  - Could end with abandoning the use case or resulting in failure
  - These should be prioritized
- **Possible conditions**
  - Alternate success path
  - Incorrect behavior by primary actor
  - Lack of response by actor or system
  - Incorrect response by actor
  - Others…..



Courses of action

- **Normal Course of action**
  - Happy path
  - Delivers the actor's goal
- **Alternate Course(s)**
  - Deliver the actor's goal but in some other way
- **Exception Course(s)**
  - Do not deliver the actor's goal



Conditions for Extensions

- **The condition should be what was detected not the reason for the condition**
  - E.g. “Wrong PIN” not “customer forgot PIN”
- **List all the conditions and start the extension scenario on the next line**

7.a. Wrong PIN:

1. ATM displays message (e.g. “wrong PIN”).
2. Return to step 5

- **Merge equivalent conditions if the extension scenario is the same**
  - E.g. “Card Damaged”, “Card upside down”, “Not an ATM card” are all the same as “ATM cant read card”

Extract and Merge

Extract complex flows, merge trivial flows

- **Create sub-use cases when**
  - A set of actions is repeated in more than one use case
  - The use case is too hard to read
    - More than a couple pages
    - Extensions have too many levels of indentation (3 or so)
  - There are more than a dozen or so steps
- **Merge use cases when**
  - The standalone use case provides no value
- **Reference other use cases when appropriate**
  - E.g. customer saves the report (UC-44: Save Report)

Tips for effective use cases

- **Be Productive, Not Perfect**
  - Developing use cases is an iterative process
  - Don’t strive for perfection out of the gate
- **Standardize syntax of action steps**
  - First word is always and actor
  - Second word is always an action
  - Remaining words are whatever is needed to further describe the action

More Tips

- **Describe the happy path first, assume success**
  - Add extensions later
  - Avoid the extreme edge use cases if possible
- **Make them easy to read**
- **Identify “who has the ball” at each step**
- **Keep UI design out of use cases**

Common Mistakes

- Use case doesn’t show system actions/responses
- Cant tell who has the ball
- Goals that are too low….. Too much detail in use case
- Requirements mixed in the use case
- Use case is not understood by everyone

Example User Level Use Case

UC-1: Access ATM

Primary actor: Customer

Description: This use case allows bank customers to log in to an ATM terminal in order to access remote banking services. This use case describes the process for an ATM in which the card is swiped (not taken).

Pre-conditions:

Customer has a card of the correct form factor to fit the card reader.

Customer has an **Account with a PIN.**

ATM is online and in service.

Bank Welcome Screen is displayed, with instruction to swipe card.

Success guarantees:

Customer is authenticated**.**

Menu of Services is displayed.

Trigger:

Main success scenario:

1. Customer swipes card.
2. ATM prompts for language.
3. Customer selects language.
4. ATM reads **card.**
5. ATM prompts for PIN.
6. Customer enters PIN.
7. ATM validates PIN.
8. ATM Displays Menu of Services.

Extensions:

***.a. At any time during the transaction, Customer selects "cancel" option:**

1. ATM displays message (e.g. "are you sure?").
2. Customer selects "yes" option.
3. ATM Displays Bank Welcome Screen.

4.a. ATM can't read card:

1. ATM displays message (e.g. "can't read card").
2. ATM displays prompt to re-swipe card.
3. Return to step 1.

7.a. Wrong PIN:

1. ATM displays message (e.g. "wrong PIN").
2. Return to step 5.

7.b. Wrong PIN 3 times:

1. ATM displays message (e.g. "there's been a problem…").
2. ATM sends message to bank to lock account.
3. ATM Displays Bank Welcome Screen.

No Requirements Here
No button pushing detail
No Screen Designs

Use Cases & Test Cases

- **Use Case are the foundation for test cases**
- **Use Cases easily turn into test plans**
- **The relationship :**

Use cases…

- Provide a functional description of the system
- Document expected behavior of the system in various scenarios
- Allow derivation of test cases
- Are written by BAs

Test cases…

- Provide a description of the tests of the system
- Document how satisfactory system behavior will be determined
- Can reveal flaws in use cases
- Are written by testers

Questions?