Use Case Descriptions

**Key Ideas**

* Illustrates the activities that are performed by users of a system
* They are logical models they describe the activities of a system without specifying how they are implemented. (What not How)

**Use Case Descriptions**

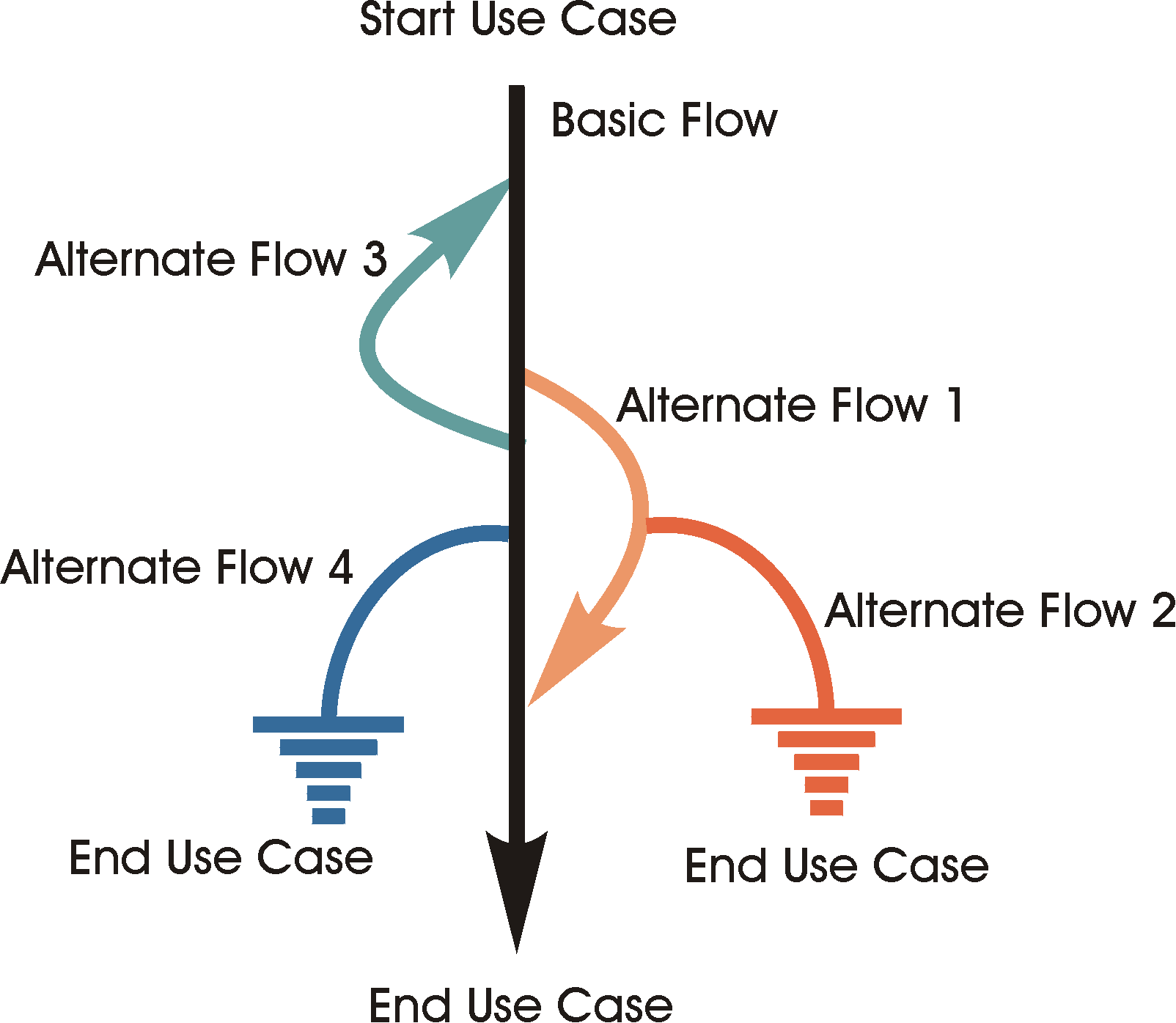
* Describes the basic functions of the system
  + What the user can do
  + How the system responds
* Building blocks for continued design activities
* **“The use case starts when”**
* **“The system does this… the actor does that”**
* **“The system does this…, the use case ends”**

**Creating Use Case Descriptions**

* Two Steps:
  + Write text-based description
  + Translate into a diagram
* Describes only one function but can have many alternative paths

**Types of Use Cases**

* Overview vs Detail
  + **Overview:** Defines high level requirements like the name, actors, and a brief description
  + **Detail:** Detail of all the information needed for the case like primary flow, alternate flow, etc.)
* Essential vs Real
  + **Essential:** Only providing the bare minimum to understand the functionality
  + **Real:** Going into further detail to describe specific sets of steps.

**Use Case Format**

1. Use Case Name
2. Description
   1. Verb-Noun format
   2. State what it is and why it exists
   3. Only needs to be a few sentences
   4. **Bolded words are in the glossary**
3. Actors
4. Pre-conditions
   1. Conditions that must be true about the state of the actor and/or system BEFORE the use case can start
   2. They do not start a use case
5. Basic flow
   1. It is the expected path (Happy path)
   2. Describes the normal way the actor interacts with the case
   3. Describe how the case ends
   4. Start by clearly defining the actor that starts the use case
6. Alternate flow
   1. Less common, optional behaviour or error conditions
   2. Cover exceptional behaviour
      1. Specific: Start at a specific point
      2. General: Can start at any point
      3. Bounded: can only occur in a range
   3. Named and numbered
   4. At {extension point} when/if <something happens>
7. Sub-flows
   1. Self-contained, named sections with their own clear purpose
   2. Complex events can be broken down
   3. The sub flows don’t need to be executed but it improves readability
   4. Needs to be named and numbered
8. Post conditions
   1. Conditions that must be true about the state of the actor and/or system AFTER the use case
9. Special requirements
   1. Requirements don’t fit nicely in a use case
   2. They are often in a separate document but can be in a special section of the use case
   3. Types of supplementary requirements (basically non-functional requirements)
      1. Legal & regulatory
      2. Quality, usability, reliability

**Prioritizing Use Cases**

* High, Medium, Low depending on:
  + Importance to business process
  + Complex, risky or time critical functionality
  + Technology needed to support needs research
* **Higher number = higher priority (0-5)**

**Defining Flow of Events**

* Start with “The use case starts when the…”
* End with “The use case ends”
* DON’T describe the user interface
  + Good: “When the customer selects to browse”
  + Bad: “When the customer clicks the link to browse”
* Start every action with “The actor” or “The system”
* Answer “What” not “How”
* Describe things so an outsider can understand them.
* Keep it simple
* **Things to avoid**
  + **Adverbs:** very, more, rather
  + **Vague terminology:** Information, appropriate, required, relevant
  + **Compound statements:** and…if….then….and….where…..and…..therefore
* Use correct punctuation
* Make glossary terms bold

**Extension Points {this is the format}**

* Named places in the flow of event where additional behaviour can happen
* No naming conventions but make it clear
* Put them in the left margin or their own line
* Can be used once
  + Example: {display product catalogue}
* Or multiple times
  + Example: {out of stock}

**Glossary**

* Supporting documentation for a use case description
* A set of shared definitions of things in the problem domain
* Domain model

**Some guidelines**

* Write in an active voice (“the system validates” over “the amount should be validated”)
* Write in present tense (“The system validates” not “the system will validate”)
* Write in newspaper style (simple, top down sentences)
* Ensure a sensible set of steps
* Use KISS principle