Recall The Team Skills

1. Analyzing the Problem (with 5 steps)
2. Understanding User and Stakeholder Needs
    1. Interviews & questionnaires
    2. Workshops
    3. Brainstorming and idea reduction
    4. Storyboarding
3. **Defining the System**
4. Managing Scope
5. Refining the System Definition
6. Building the Right System

Team Skill 3: Defining the System

- Define the system by focusing more on the features and deal with the increased amount of information.

- Ch 14: A Use Case Primer
- Ch 15: Organizing Requirements Information
- Ch 16: The Vision Document

- Ch 17: Product Management (skipped)

Chapter 14
A Use Case Primer

- Use cases basics
- Benefits of use cases
- Steps of building use case model
- Use cases, storyboarding, and user interface design

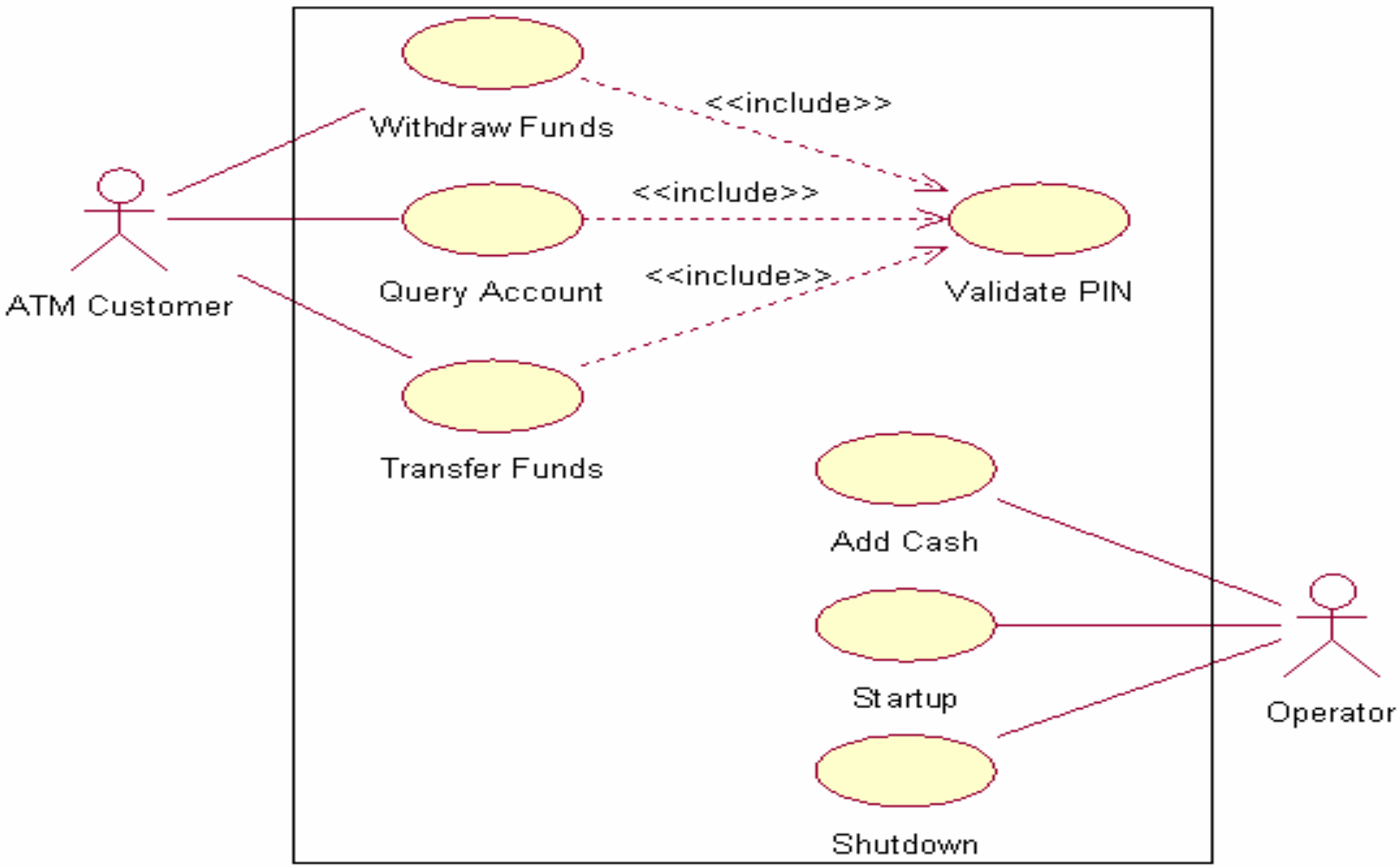
Use Case Basics

- A use case describes a sequence of actions the system performs that yield an observable result of value to a particular actor.
- An actor is someone or something that interacts with the system.
  - Users, other systems, or devices
- Documenting use cases by using
  - use case templates: Name, decscriptions,
  - Covered in the labs (UML Notes)

Use Case Basics

- Documenting use cases (Covered in the labs) by using
- use case templates:
  - Name
  - Description
  - Actor(s)
  - Flow of events
  - Pre-conditions
  - Post-conditions
  - Etc ...
- UML Diagrams

Example


Withdraw Funds
<<include>>
ATM Customer
<<include>>
Query Account
Validate PIN
<<include>>
Transfer Funds
Add Cash
Startup
Operator
Shutdown

The Benefits of Use Cases

- They are relatively easy to write and easier to read.
- They force developers to think through the design of a system from the perspective of a user.
- They engage the users in the requirements process:
  - helping them understand the system that is being proposed.
  - giving them a way to communicate and document their needs.
- They give context for the requirements of the system:
  - One can understand why a requirement is what it is
  - as well as how the system meets its objectives.

The Benefits of Use Cases

- They provide an ordering mechanism for requirements:
  - one can tell what has to happen before the next thing happens, and so on.
- In most circumstances, developers write the use cases. That means not only that there actually are understood requirements but also that the developers know they are responsible for determining them.
- It is a critical tool in the analysis process, helping us understand what the system needs to do and how it might go about doing it.

The Benefits of Use Cases

- It is a critical tool in the design and implementation process, reducing the risk of transitioning from an expression of requirements to a differing implementation
- They carry over directly into the testing process, helping to assure that the system actually does what it was intended to do
- They serve as inputs to the user documentation, conveniently organized in a step-by-step format.

Use Case Model

- Individual use case describes how a particular actor interacts with the system to achieve a result of value to the specific actor.
- The set of all use cases together describes the complete behavior of the system.
- The complete set of use cases, actors, and their interactions constitutes the use-case model for the system.

Building a uses case model

- write individual use cases and then add them all .. Not a good idea.
- Instead, build a context model of the system and successively refine it.
- That's better for understanding, communicating, and refining the behavior of the system in an iterative development.

Building the Use-Case Model

- Step 1: Identify and Describe the Actors
- Step 2: Identify the Use Cases and Write a Brief Description
- Step 3: Identify the Actor and Use-Case Relationships
- Step 4: Outline the Individual Use Cases
- Step 5: Refine the Use Cases

Step 1: Identify and Describe the Actors

- Who uses the system?
- Who gets info from the system?
- Who provides info to the system?
- Where in the company is the system used?
- Who supports the and maintain the system?
- What other systems use this system?

Step 2: Identify the Use Cases and Write a Brief Description

- Typically, the use-case name is a few words or a short phrase that starts with an action verb and communicates what the actor achieves with the use case.
- To identify the use cases, ask the following for each actor:

Step 2: Identify the Use Cases and Write a Brief Description

- What will the actor use the system for?
- Will the actor create, store, change, remove, or read data in the system?
- Will the actor need to inform the system about external events or changes?
- Will the actor need to be informed about certain occurrences in the system?

Step 3: Identify the Actor and Use-Case Relationships

- Only one actor can initiate a use case
- However, many actors can be involved in a use case.
- Each use case is analyzed to see what actors interact with it and
- Each actor's behavior is reviewed to make sure that all of the results he needs to see are achieved the system.
- If this becomes complex, uses diagrams.

Step 4: Outline the Individual Use Cases

- To understand the system more think about the flow of events (basic and alternatives) for each use case.
- Basic flow: the most common path from start to finish through the system
- Alternative flows: other possible paths based on regular or exceptional circumstances.

Step 4: Outline the Individual Use Cases

- To do that ask the following questions:
- Basic flow:
  - What events start the use case?
  - How does the use case end?
  - How does the use case repeat some behavior?
- Alternative Flows:
  - Are there optional situations in the use case?
  - What odd cases might happen?
  - What variants might happen?
  - What may go wrong? ..etc

Step 5: Refine the Use Cases

- When to refine and think about the next level of details?
- Depends on
  - All alternative flows including exception conditions
  - Pre- and post-conditions

Use Cases and User Interfaces

- User interface design and use-case specification tend to be parallel.
- For example, inside the context of a specific use case, the team must decide:
  - What choices the user makes given the screens, dialog boxes, and so on that we have presented to the user,
  - what the system does based on those selections, and
  - what screens, choices, and so on the system presents to the user.

Use Cases and User Interfaces

- In other words, each step in a use case is achieved via the presentation of a GUI of some kind,
- followed by a user selection of some kind,
- followed by a presentation of a new GUI that moves the user to the next system context, and so on.

Use Cases and Storyboarding

- A Use Case Storyboard Example

Use Case Sequence: Inserting Online Clip Art

(This series of steps within a larger use case allows the user to access an online clip art repository and select and insert a new clip art item.)

1. The user puts the cursor at the desired clip art location and selects the function for inserting clip art.
2. The system displays the clip art source locations.
3. The user selects the "Clips Online" choice.
4. The system launches the browser and automatically navigates to the online library
5. The user navigates to the selected art item.

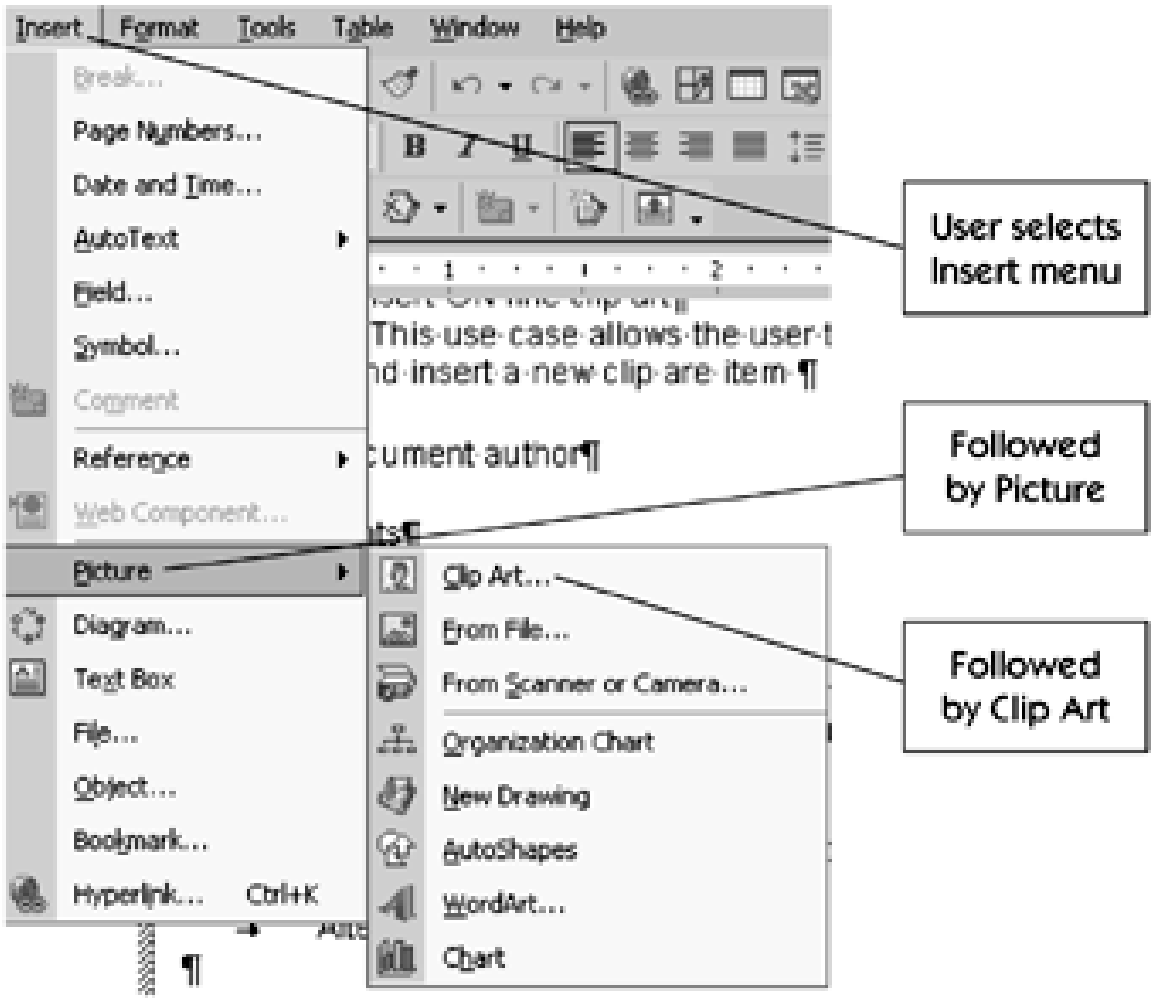
. . .

Use Cases and Storyboarding

- You can use Microsoft PowerPoint as your storyboard presentation tool to build one PowerPoint slide for each of the steps in the use case to show the user how you intend the system to work.
- In this fashion, you can develop the use case and the GUIs in parallel while involving the user in the concept review process at the same time.

Storyboard slide for step 1 of a use case

1. **Select function for inserting clip art**



Storyboard slide for step 4 of a use case

4. **System launches browser and automatically navigates to remote library**

Reading Assignment

- Read the case study of the HOLIS use cases pages: 159-163

Key Points

- Use cases carry the majority of the requirements for the system.
- The development team, with user involvement, writes the use cases.
- Use cases are built on a common, standard format.
- Use cases later drive test case development.