Software Development Project (SWP391)

Software Requirement Guides

Agenda

- Requirement Overview
- Use Cases & Actors
- Identifying Use Cases
- Documenting Use Cases
- Business Rules
- Requirement Prototyping

Requirement Overview

- The requirements of a system describe
  - What the user expects from the system
  - What the system will do for the user
- When defining the requirements of a system
  - The system should be viewed as a black box. Only the external characteristics of the system are considered
  - Both functional & nonfunctional requirements need to be considered
- Requirements modeling consists of ***requirements analysis*** & ***requirements specification***

Req Overview: Requirements Analysis

- The software requirements describe the functionality that the system must provide for the users.
- Requirements analysis involves analyzing the requirements by
  - Interviewing users, stakeholders
  - Analyzing the existing system(s)
    - Understanding and documenting the current system
    - Determining which features of the current system should be automated and which should remain manual
    - Discussing with users what functions could be done differently when the system is automated.

Req Overview: Requirements Specification

- The document that needs to be agreed on by the requirements analysts and the users.
  - Starting point for the subsequent design & development
  - Both functional requirements & nonfunctional requirements need to be specified
- A functional req. describes the functionality the system must be capable of providing in order to fulfill the purpose of the system
  - Functionality the system needs to provide
  - Input to the system from the external environment
  - Output to the external environment
  - What stored information the system reads or updates
- A nonfunctional req: quality attribute, or quality-of-service goal that the system must fulfill

Use Cases & Actors

The ***use case model*** describes the functional requirements of the system in terms of the actors & use cases

- The system is treated as a black box (dealing with ***what*** the system does in response to the actor's inputs)
- Functional requirements are described in terms of actors, which are users of the system, and use cases

An ***actor*** provides inputs to the system and the system provides responses to the actor

A ***use case (UC)*** defines a sequence of interactions between one or more actors and the system

- A use case always starts with input from an actor
- Typically consists of a sequence of interactions between the actor and the system
- Each interaction consists of an input from the actor followed by a response from the system

Use Cases & Actors
What are actors?

An actor characterizes an external user (i.e., outside the system) that interacts with the system

- The only external entities that interact with the system
- Actors are outside the system and not part of it
- A user is an individual, whereas an actor represents the role played by all users of the same type
- There are other types of actors in addition to or in place of human actors: external systems, I/O devices, or timers

Use Cases & Actors
Sample actors

- Users
  - Homeowners are actors on HOLIS system
  - Authors are actors on a word processing system
- Other systems or applications
  - HOLIS Control Switch is an actor on the HOLIS Central Control Unit
- A device
  - Lights
  - Printer

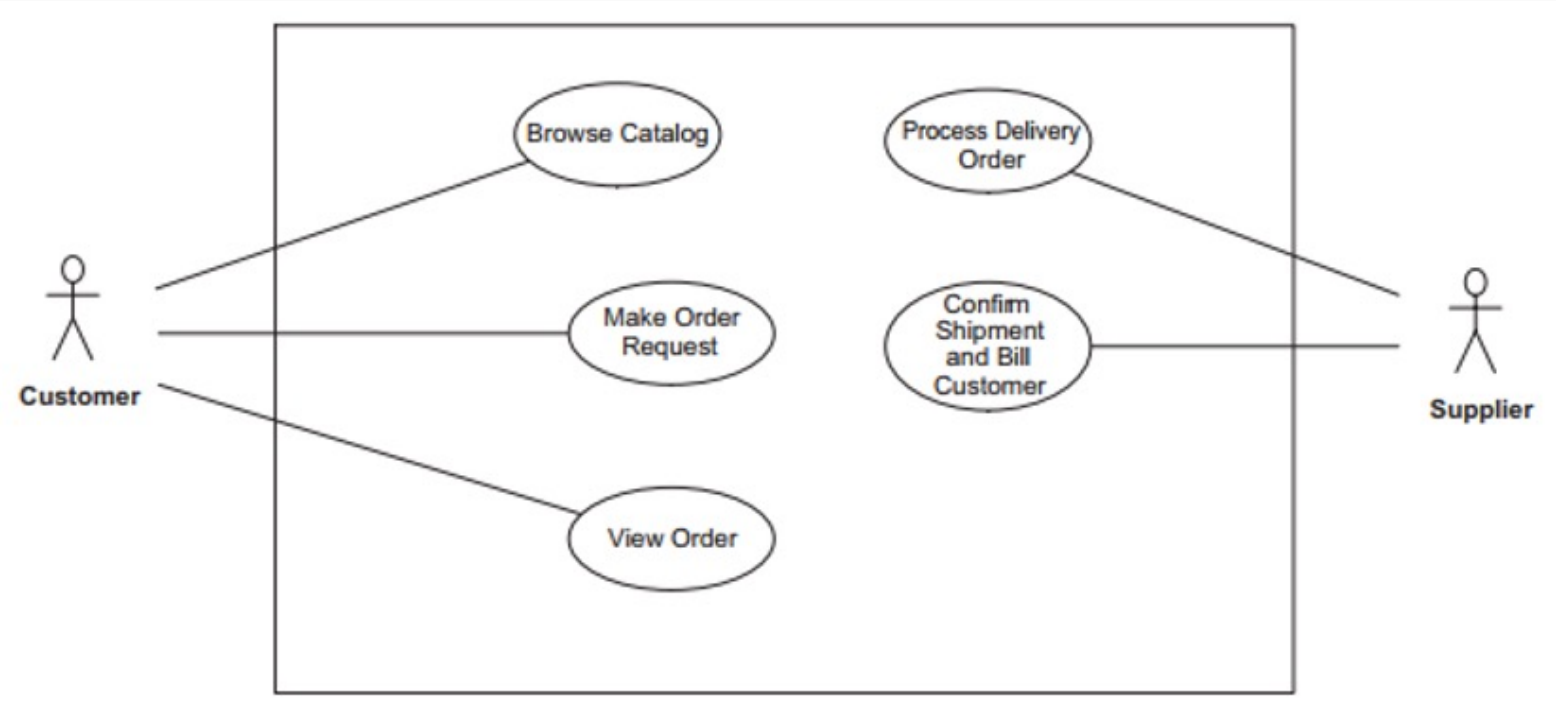
Use Cases & Actors
Identifying actors

Some questions to help identify actors:

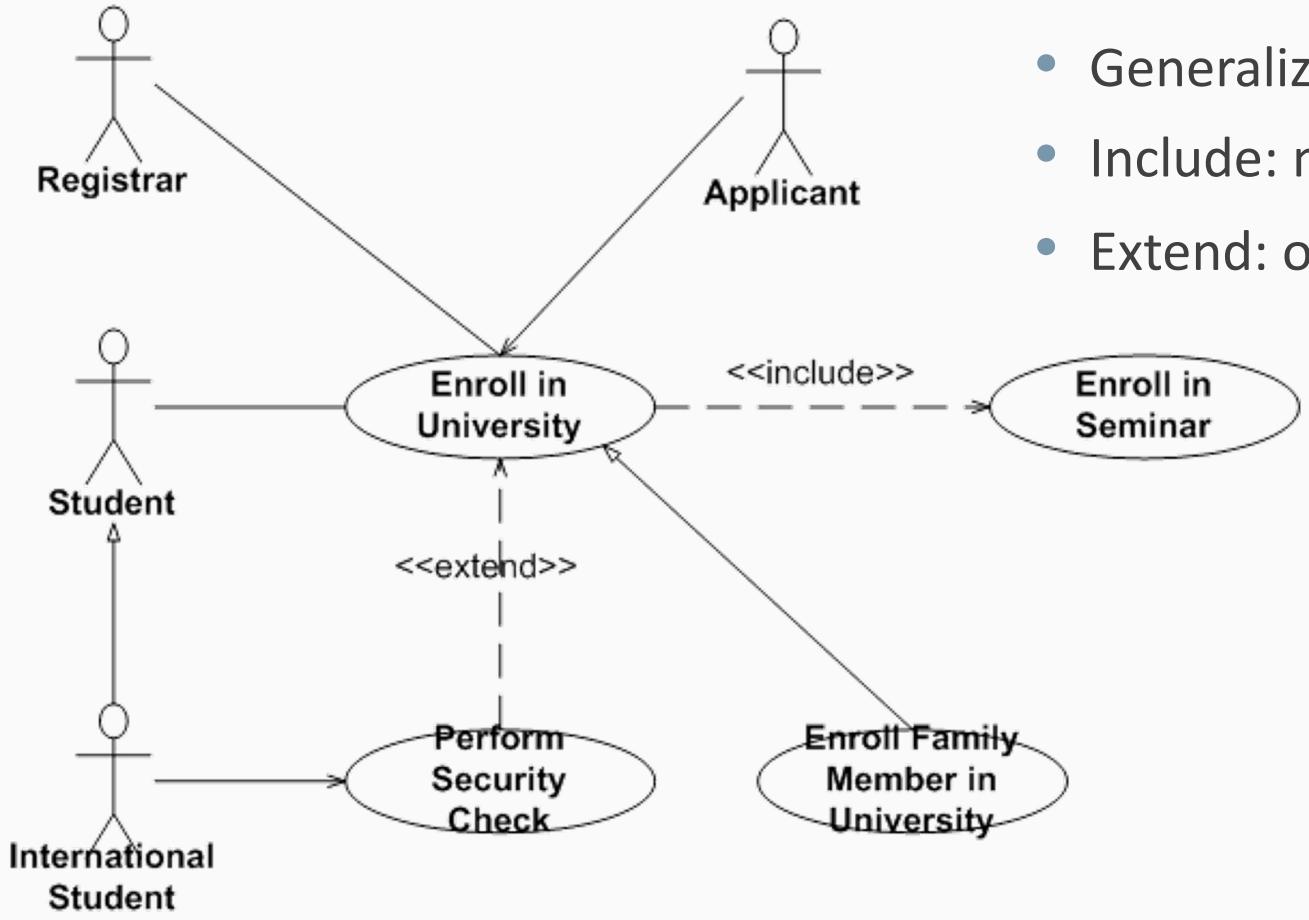
- Who (or what) **is notified** when something occurs within the system?
- Who (or what) **provides information** or services to the system?
- Who (or what) **helps the system** respond to and complete a task?

Use Cases & Actors
Simple online shop example

The Online Shopping System is a highly distributed Web-based system that provides services for purchasing items such as foods, books or clothes.



Use Cases & Actors
UC Diagram & Relationships



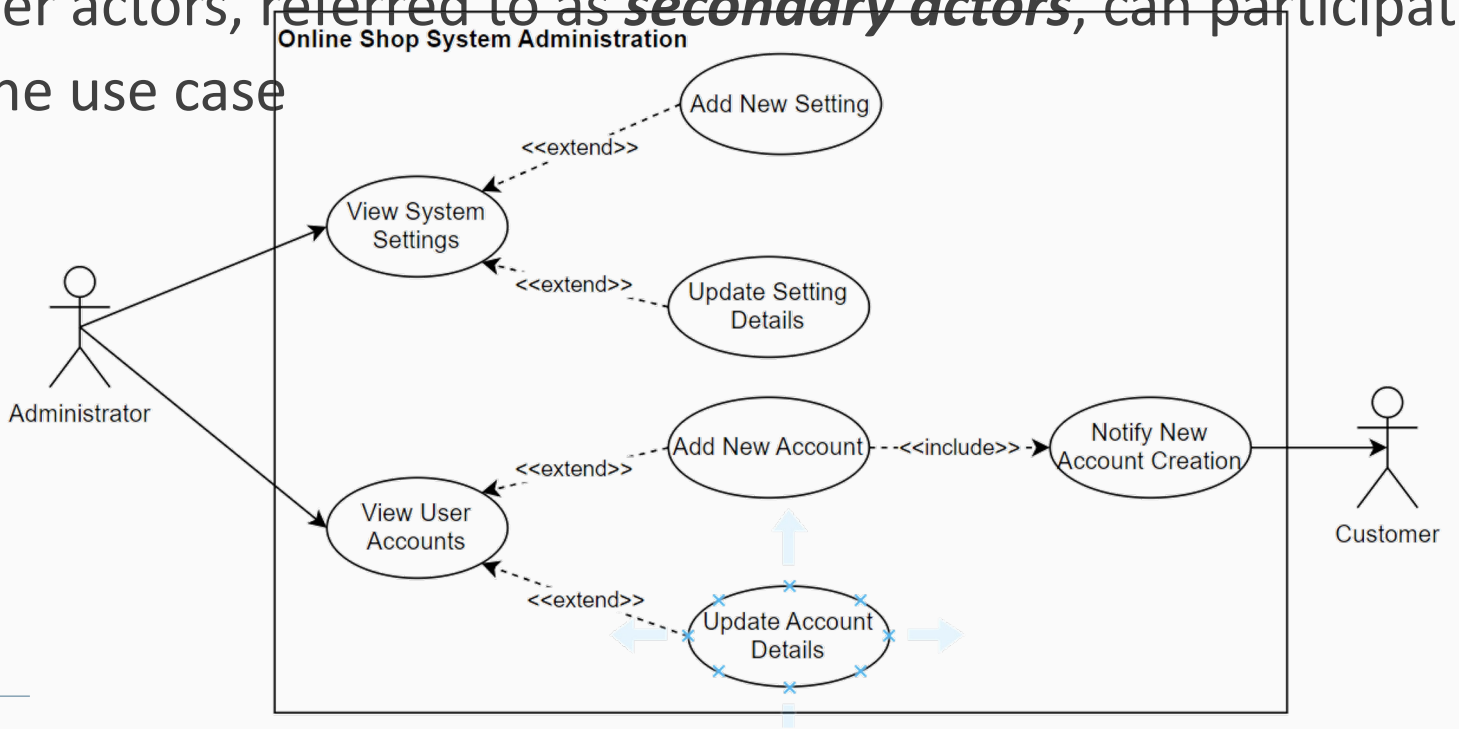
UC Relationships

- Generalization
- Include: mandatory
- Extend: optional

Use Cases & Actors
Primary vs secondary actors

A ***primary actor*** initiates a UC => UC starts with an input from the primary actor to which the system has to respond (gain value from the UC)

Other actors, ~~referred to as ***secondary actors***, can participate~~ in the use case



Identifying Use Cases

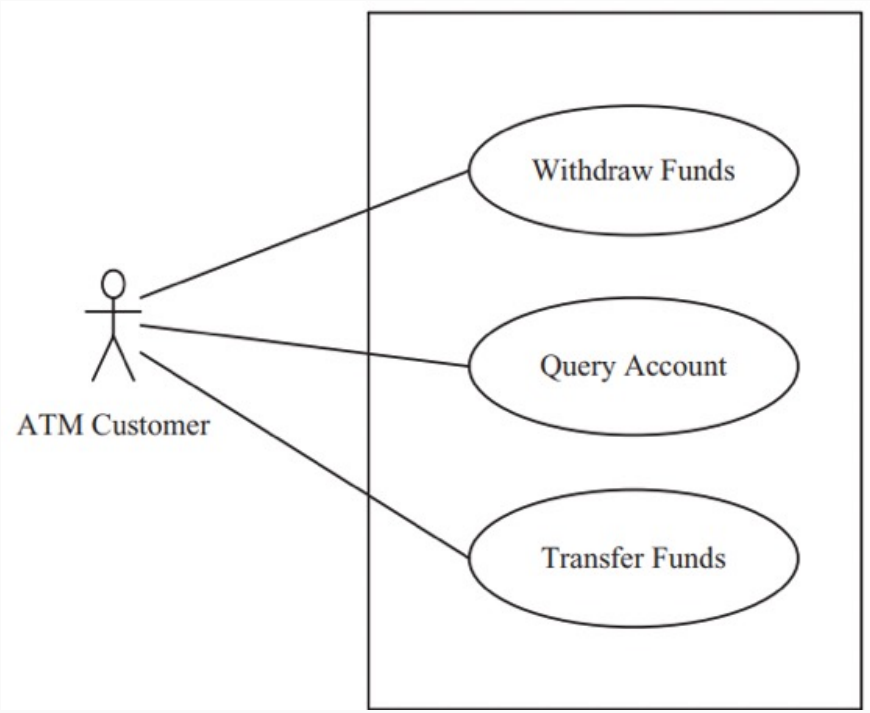
- A use case (UC) describes a sequence of interactions between a system and an external actor that results in the actor being able to achieve some outcome of value
- In this way, the functional requirements of the system are described in terms of the UCs, which constitute functional specification of a system.
- To determine the UCs in the system, it is useful to **start by considering the actors & the interactions they have with the system.**
- When developing UCs, it is important to avoid a functional decomposition in which several small UCs describe small individual functions of the system rather than describe a sequence of events that provides a useful result to the actor

Identifying Use Cases
Use case considering and naming

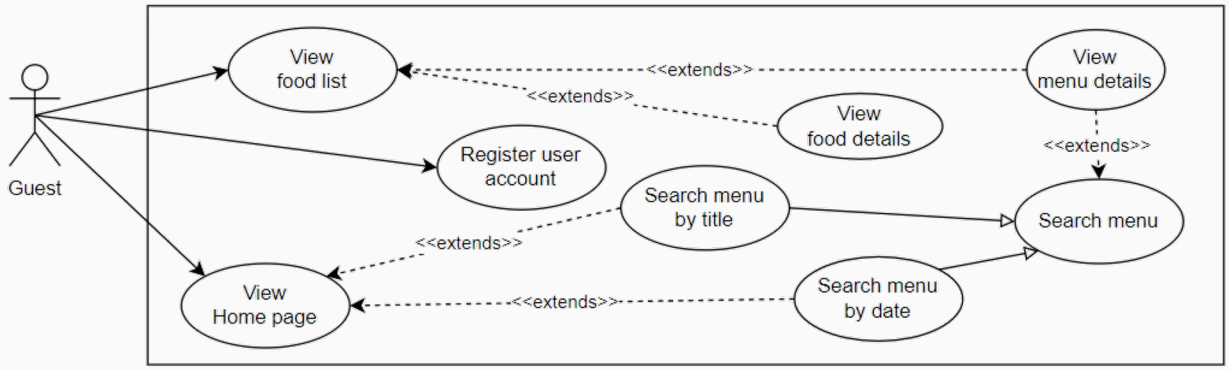
- Questions to consider when identifying use cases
  - What will the actor use the system for?
  - Will the actor create, store, change, remove, or read data in the system?
  - Will the actor need to inform the system about external events or changes?
  - Will the actor need to be informed about certain occurrences in the system?
- Names of UCs are always ***written in the form of a verb followed by an object***.
- Select strong, descriptive names to make it evident from the name that the use case will deliver something valuable for some user.

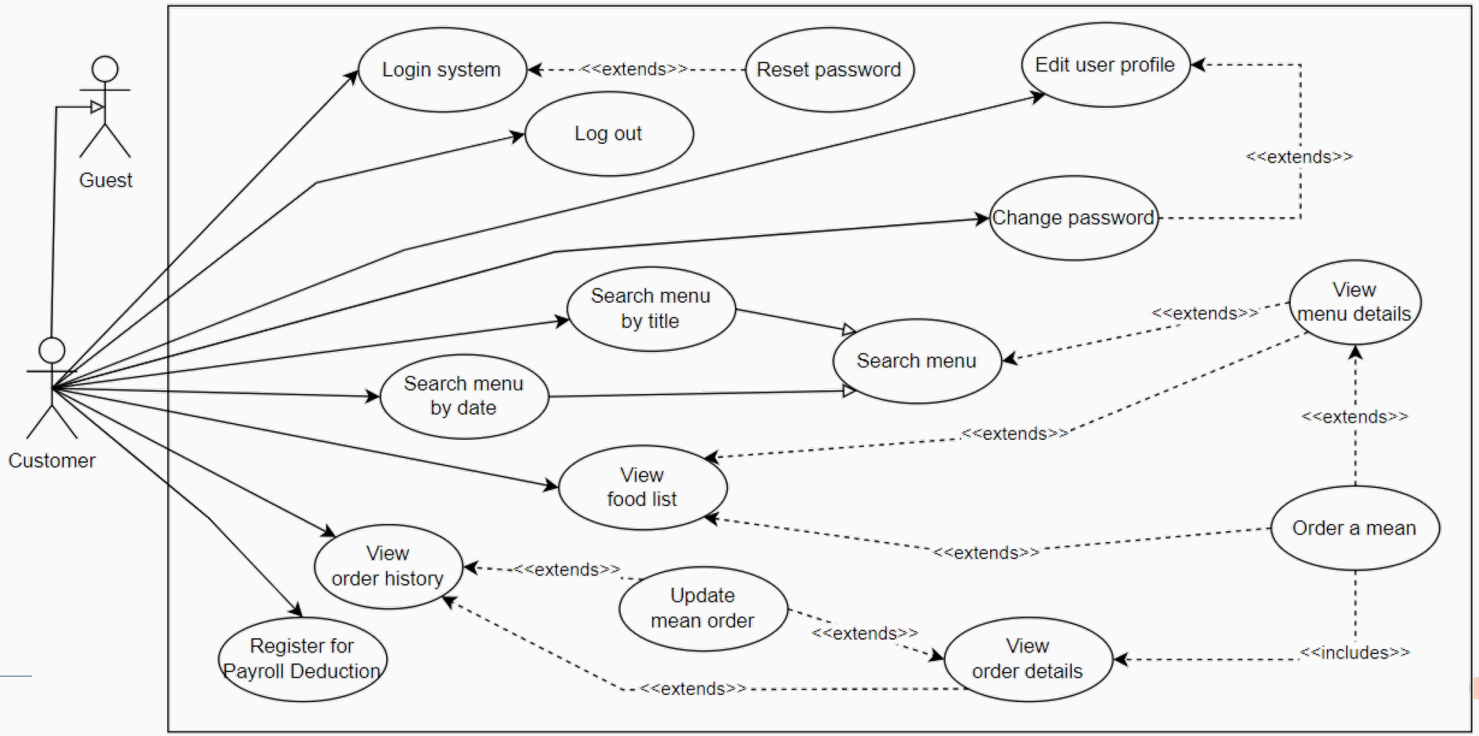
Identifying Use Cases
Simple Banking Example

- In addition to withdrawing cash from the ATM, the ATM Customer actor is also allowed to query an account or transfer funds between two accounts.
- Because these are distinct functions initiated by the customer with different useful results, the query and transfer functions should be modeled as separate UCs, rather than being part of the original UC



Identifying Use Cases
Online Shopping Example





Documenting Use Cases

UC Name:			
Primary Actor:		Secondary Actors:	
Trigger:			
Description:			
Preconditions:			
Postconditions:			
Normal Flow:			
Alternative Flows:			
Exceptions:			
Business Rules:			

Trigger
Identify the business event, system event, or user action that initiates the use case. This trigger alerts the system that it should begin testing the preconditions for the use case so it can judge whether to proceed with execution.

Preconditions
List any activities that must take place, or any conditions that must be true, before the use case can be started.

Postconditions
Describe the state of the system at the successful conclusion of the use case execution.

Normal Flow
Provide a description of the user actions and corresponding system responses that will take place during execution of the use case under normal, expected conditions.

Alternative Flows
Document other successful usage scenarios that can take place within this use case. State the alternative flow, and describe any differences in the sequence of steps that take place.

Exceptions
Describe any anticipated error conditions that could occur during execution of the use case and how the system is to respond to those conditions.

Business Rules
List any business rules that influence this use case. Don't include the business rule text here, just its identifier so the reader can find it in another repository when needed.

Documenting Use Cases

UC Spec

- Use case name
- Actors
- Summary
- *Dependency*
- Preconditions
- Main sequence
- Alternative sequences
- Postcondition

Use case name: Withdraw Funds

Actor: ATM Customer

Summary: Customer withdraws a specific amount of funds from a valid bank account.

Dependency: Include Validate PIN use case.

Precondition: ATM is idle, displaying a Welcome message.

Main sequence:

1. Include Validate PIN use case.
2. Customer selects Withdrawal, enters the amount, and selects the account number.
3. System checks whether customer has enough funds in the account and whether the daily limit will not be exceeded.
4. If all checks are successful, system authorizes dispensing of cash.
5. System dispenses the cash amount.
6. System prints a receipt showing transaction number, transaction type, amount withdrawn, and account balance.
7. System ejects card.
8. System displays Welcome message. **Alternative sequences**:

Step 3: If the system determines that the account number is invalid, then it displays an error message and ejects the card.

Step 3: If the system determines that there are insufficient funds in the customer's account, then it displays an apology and ejects the card.

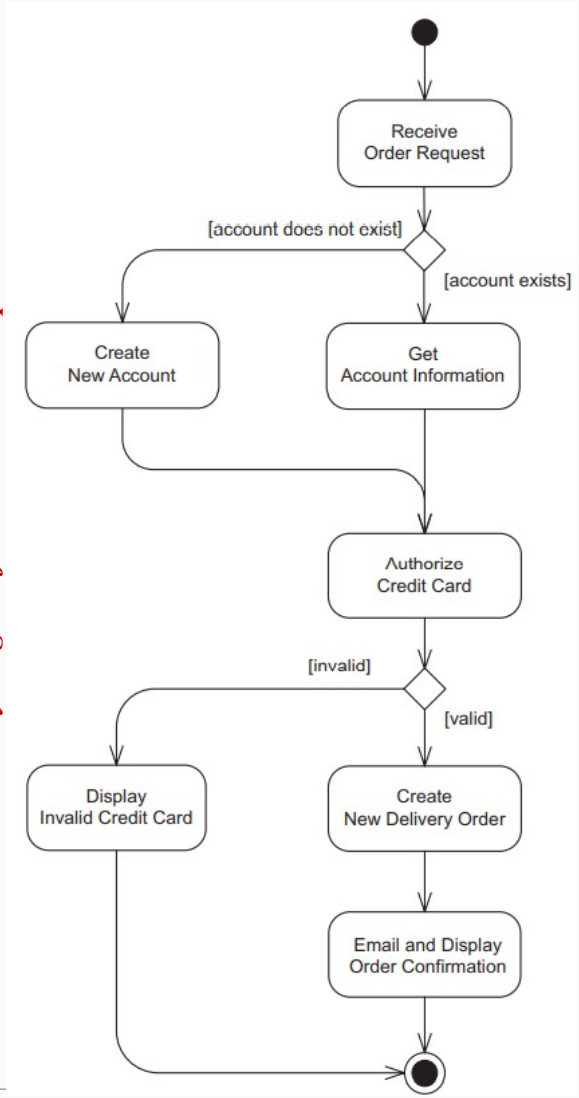
Step 3: If the system determines that the maximum allowable daily withdrawal amount has been exceeded, it displays an apology and ejects the card.

Step 5: If the ATM is out of funds, the system displays an apology, ejects the card, and shuts down the ATM.

Postcondition: Customer funds have been withdrawn.

Documenting Use Cases

Activity diagram for Make Order Request use case



Use Case Description for Make Order Request

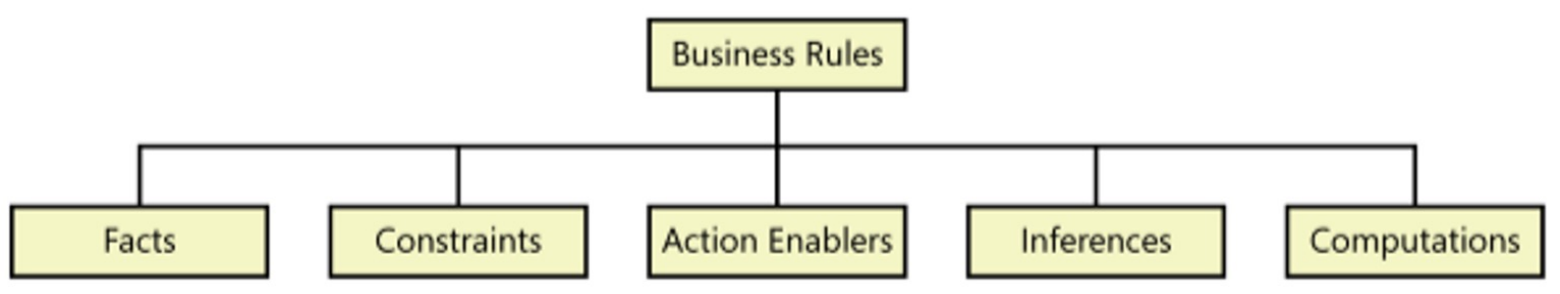
Use case name: Make Order Request
Summary: Customer enters an order request to purchase catalog items. The customer's credit card is checked for validity and sufficient credit to pay for the requested catalog items.
Actor: Customer
Precondition: Customer has selected one or more catalog items
Main sequence:

1. Customer provides order request and customer account Id to pay for purchase.
2. System retrieves customer account information, including the customer's credit card details.
3. System checks the customer's credit card for the purchase amount and, if approved, creates a credit card purchase authorization number.
4. System creates a delivery order containing order details, customer Id, and credit card authorization number.
5. System confirms approval of purchase and displays order information to customer.

Alternative sequences:
Step 2: If customer does not have an account, the system prompts the customer to provide information in order to create a new account. The customer can either enter the account information or cancel the order.
Step 3: If authorization of the customer's credit card is denied (e.g., invalid credit card or insufficient funds in the customer's credit card account), the system prompts the customer to enter a different credit card number. The customer can either enter a different credit card number or cancel the order.
Postcondition: System has created a delivery order for the customer.

Business Rules

- Statements that define or constrain some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business



ID	Category	Rule Definition
BR-01	Constraints	Delivery time windows are 15 minutes, beginning on each quarter hour
BR-02	Constraints	Deliveries must be completed between 10:00 A.M. and 2:00 P.M. local time, inclusive.
BR-03	Facts	All meals in a single order must be delivered to the same location.
BR-04	Facts	All meals in a single order must be paid for by using the same payment method.
BR-11	Constraints	If an order is to be delivered, the patron must pay by payroll deduction.
BR-12	Computations	Order price is calculated as the sum of each food item price times the quantity of that food item ordered, plus applicable sales tax, plus a delivery charge if a meal is delivered outside the free delivery zone.

Business Rules
Business Rules Taxonomy - Facts

Facts are simply statements that are true about the business at a specified point in time. A fact describes associations or relationships between important business terms.

Examples of facts include the following:

- Every chemical container has a unique bar code identifier.
- Every order has a shipping charge.
- Sales tax is not computed on shipping charges.
- Nonrefundable airline tickets incur a fee when the purchaser changes the itinerary.
- Books taller than 16 inches are shelved in the library's Oversize section.

Business Rules

Business Rules Taxonomy - Constraints

A constraint is a statement that restricts the actions that the system or its users are allowed to perform. Someone describing a constraining business rule might say that certain actions ***must*** or ***m***

rc

Organizational policies

- A loan applicant who is less than 18 years old must have a parent or a legal guardian as cosigner on the loan.
- A library patron may have a maximum of 10 items on hold at any time.
- Insurance correspondence may not display more than four digits of the policyholder's Social Security number.

Government regulations

- All software applications must comply with government regulations for usage by visually impaired persons.
- Airline pilots must receive at least 8 continuous hours of rest in every 24-hour period.
- Individual federal income tax returns must be postmarked by midnight on the first business day after April 14 unless an extension has been granted.

Industry standards

- Mortgage loan applicants must satisfy the Federal Housing Authority qualification standards.
- Web applications may not contain any HTML tags or attributes that are deprecated according to the HTML 5 standard.

Business Rules
Business Rules Taxonomy - *Action Enablers*

A rule that triggers some activity if specific conditions are true is an *action enabler*.

A person could perform the activity in a manual process. Alternatively, the rule might lead to specifying software functionality that makes an application exhibit the correct behavior when the system detects the triggering event.

Sc n

- If the chemical stockroom has containers of a requested chemical in stock, then offer existing containers to the requester.
- On the last day of a calendar quarter, generate the mandated OSHA and EPA reports on chemical handling and disposal for that quarter.
- If the expiration date for a chemical container has been reached, then notify the person who currently possesses that container.

Business Rules

Business Rules Taxonomy - Inferences

Sometimes called *inferred knowledge* or a *derived fact*, an *inference* creates a new fact from other facts. Inferences are often written in the "if/then" pattern also found in action-enabling business rules, but the "then" clause of an inference simply provides a piece of knowledge, not an action to be taken.

Some examples of inferences are:

- If a payment is not received within 30 calendar days after it is due, then the account is delinquent.
- If the vendor cannot ship an ordered item within five days of receiving the order, then the item is considered back-ordered.
- Chemicals with an LD50 toxicity lower than 5 mg/kg in mice are considered hazardous.

Business Rules

Business Rules Taxonomy - Computations

The fifth class of business rules defines *computations* that transform existing data into new data by using specific mathematical formulas or algorithms. Many computations follow rules that are external to the enterprise, such as income tax withholding formulas.

Following are a few examples of computational business rules written in text form

- The domestic ground shipping charge for an order that weighs more than two pounds is $4.75 plus 12 cents per ounce or fraction thereof.
- The unit price is reduced by 10 percent for orders of 6 to 10 units, by 20 percent for orders of 11 to 20 units, and by 30 percent for orders of more than 20 units.

Requirement Prototyping

- Purpose:
  - Clarify, complete, and validate requirements
  - Explore design alternatives
  - Create a subset that will grow into the ultimate product
- Three classes of prototype attributes
  - Scope:
    - Mock-up prototype => UX Design
    - Proof-of-concept prototype => Technical Approach
  - Future use: throwaway prototype vs. evolutionary prototype
  - Form: paper prototype vs. electronic prototype

Requirement Prototyping
Throwaway Prototypes

- Purpose: build a throwaway prototype to answer questions, resolve uncertainties, and improve requirements quality
- A wireframe is a particular approach to throwaway prototyping commonly used for custom user interface design and website design.
- You can use wireframes to reach a better understanding of three aspects of a website:
  - The conceptual requirements
  - The information architecture or navigation design
  - The high-resolution, detailed design of the pages

Requirement Prototyping
Evolutionary Prototypes

- Purpose: provides a solid architectural foundation for building the product incrementally as the requirements become clear over time.
- Evolutionary prototyping is well suited for web development projects

Typical applications of software prototypes

	Throwaway	Evolutionary
Mock-up	■ Clarify and refine user and functional requirements. ■ Identify missing functionality. ■ Explore user interface approaches.	■ Implement core user requirements. ■ Implement additional user requirements based on priority. ■ Implement and refine websites. ■ Adapt system to rapidly changing business needs.
Proof of concept	■ Demonstrate technical feasibility. ■ Evaluate performance. ■ Acquire knowledge to improve estimates for construction.	■ Implement and grow core multi-tier functionality and communication layers. ■ Implement and optimize core algorithms. ■ Test and tune performance.

Requirement Prototyping

Working with Prototypes 1/2

User class	Use case
Visitor	Get Information about the Book Get Information about the Author Read Sample Chapters Read the Blog Contact the Author
Customer	Order a Product Download an Electronic Product Request Assistance with a Problem
Administrator	Manage the Product List Issue a Refund to a Customer Manage the Email List

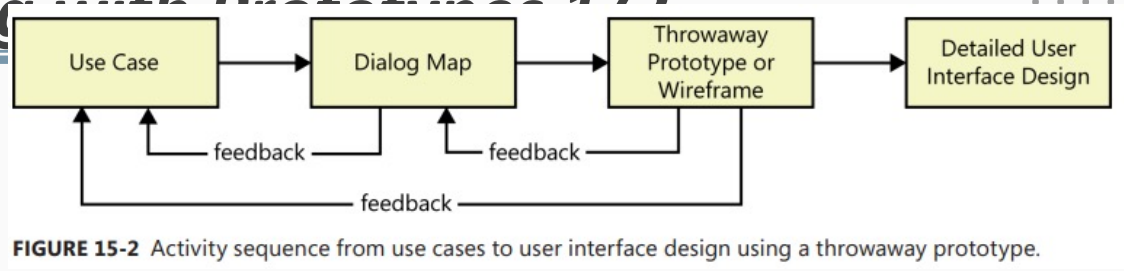
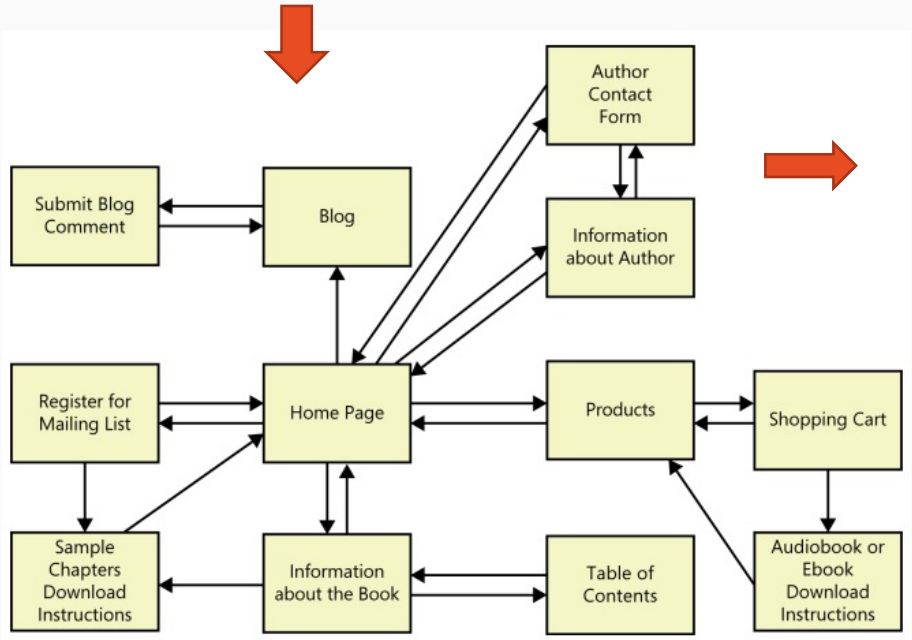
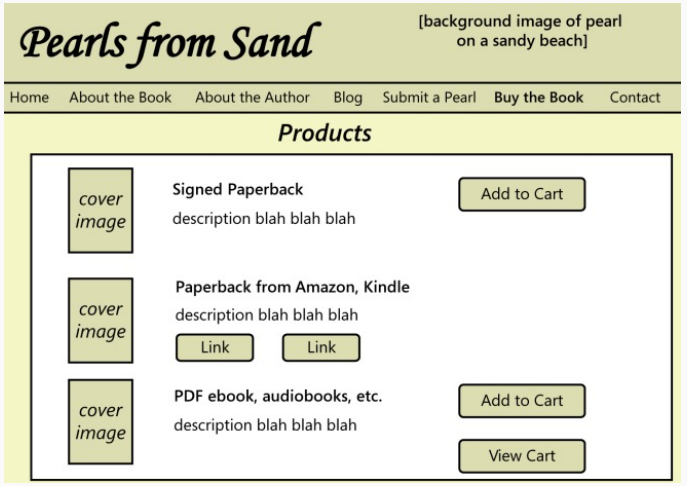


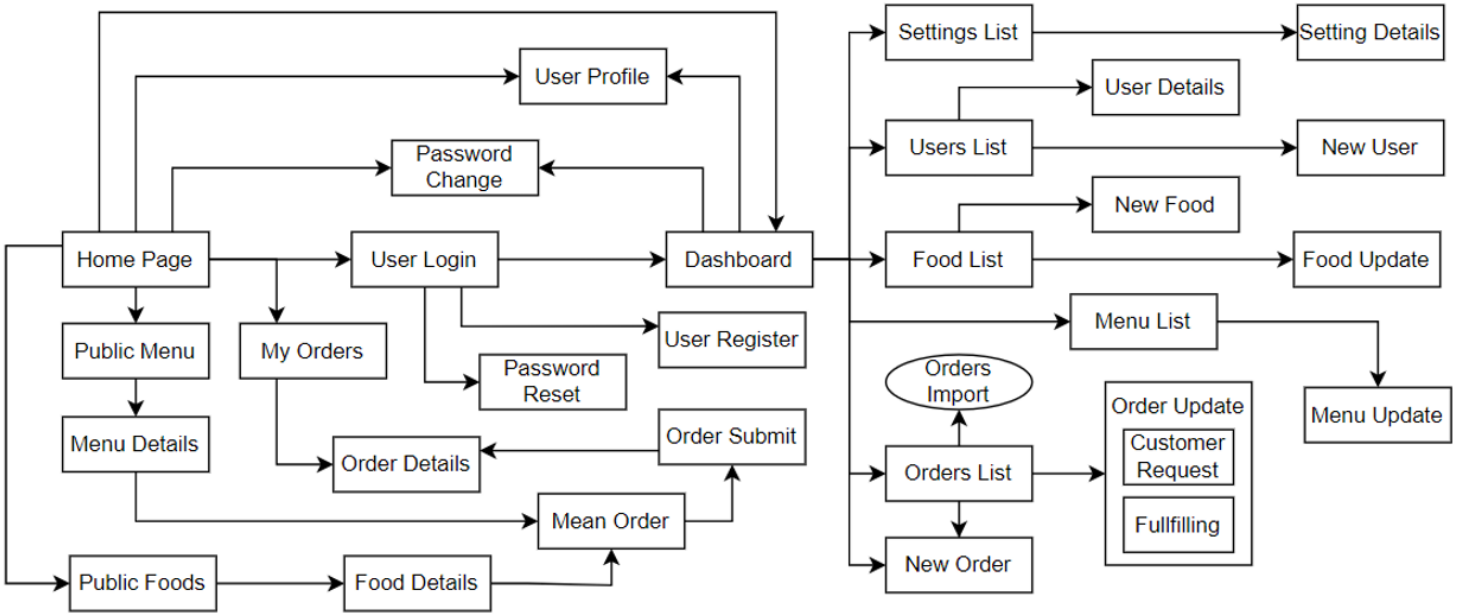
FIGURE 15-2 Activity sequence from use cases to user interface design using a throwaway prototype.





Requirement Prototyping

Working with Prototypes 2/3



2.2 Screen Descriptions

#	Feature	Screen	Description
1	Order Meals	Create Order	<<brief description of the screen>>
2	Order Meals	Change Order	..
3	..		

2.3 Screen Authorization

Screen	Role-Name1	Role-Name2	Role-Name3	...
<<Screen Name1>>	X		X	X
<<Screen Activity>>			X	X
<<Screen Name2>>	X		X	
Query All Data	X			

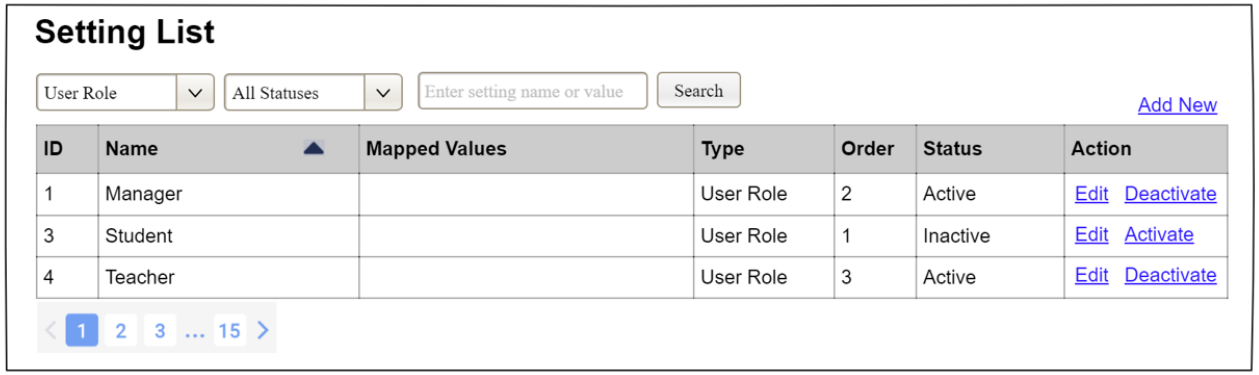
Requirement Prototyping
Working with Prototypes 3/3

Provide specifications for the screen/functions, grouped by feature, sub-featu

Description Items

1. Brief description
2. Screen layout
3. Screen Field details

This is for the administrator to view the list of current system settings. On the screen, s/he can also activate or deactivate (change status) of a specific setting.



Field Name	Field Type	Description
Filter/Search Fields		
Setting Type		Filled with the list of current active setting types Allow to filter the list by setting type; Default value is "All Types"
Setting Status		Values: All Statuses (default), Active, and Inactive Allow to filter the list by status Default value: "All Statuses"
Search Phase	String (30)	Allow to search using the name or map values Default value: blank
Search		Click to refresh the list with the defined filter(s) and search phrase.
Add New		Click to open the Setting Details page for adding new setting (master data)
Data Table		
ID	Integer	Auto-increased identifier of the setting


Q
A