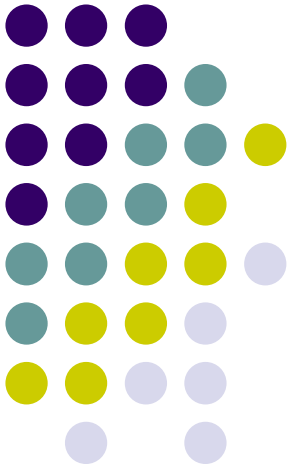
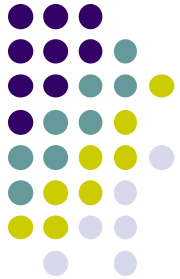
Foundations of Software Engineering
(CPSC 362)

Software Requirements

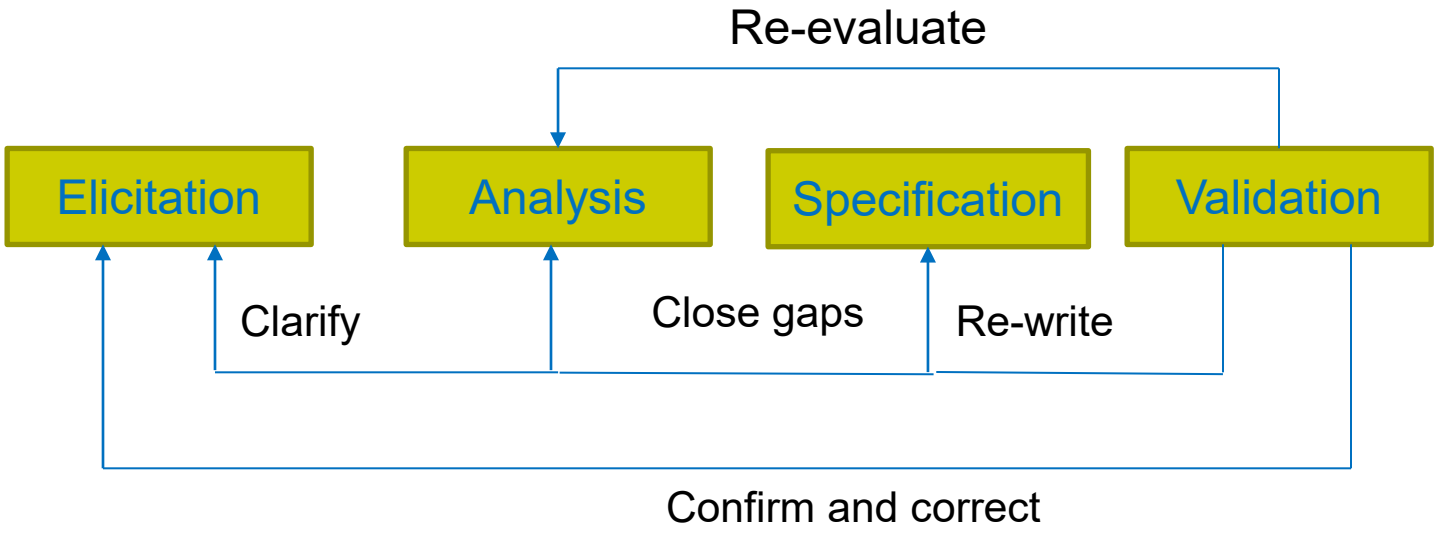


Software Requirements



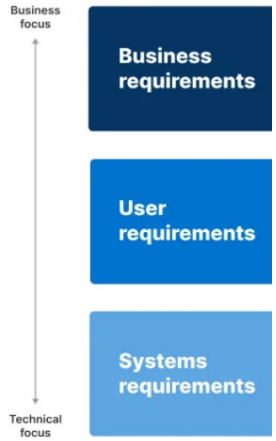
- **Requirements**
  - A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents
    - Software Requirements Specification (**SRS**) is written requirements for a software system.
- **Types of requirements**
  - Functional requirements
    - The software functionalities, satisfying the business requirements.
  - Nonfunctional requirements
    - Characteristics (quality attributes and constraints) that a system must exhibit or restriction it must respect.
      - Quality Attributes (performance, availability, scalability, efficiency, security, usability, integrity, robustness, etc.)
      - Constraints to be met (e.g., legal compliance)

Requirements Development Process

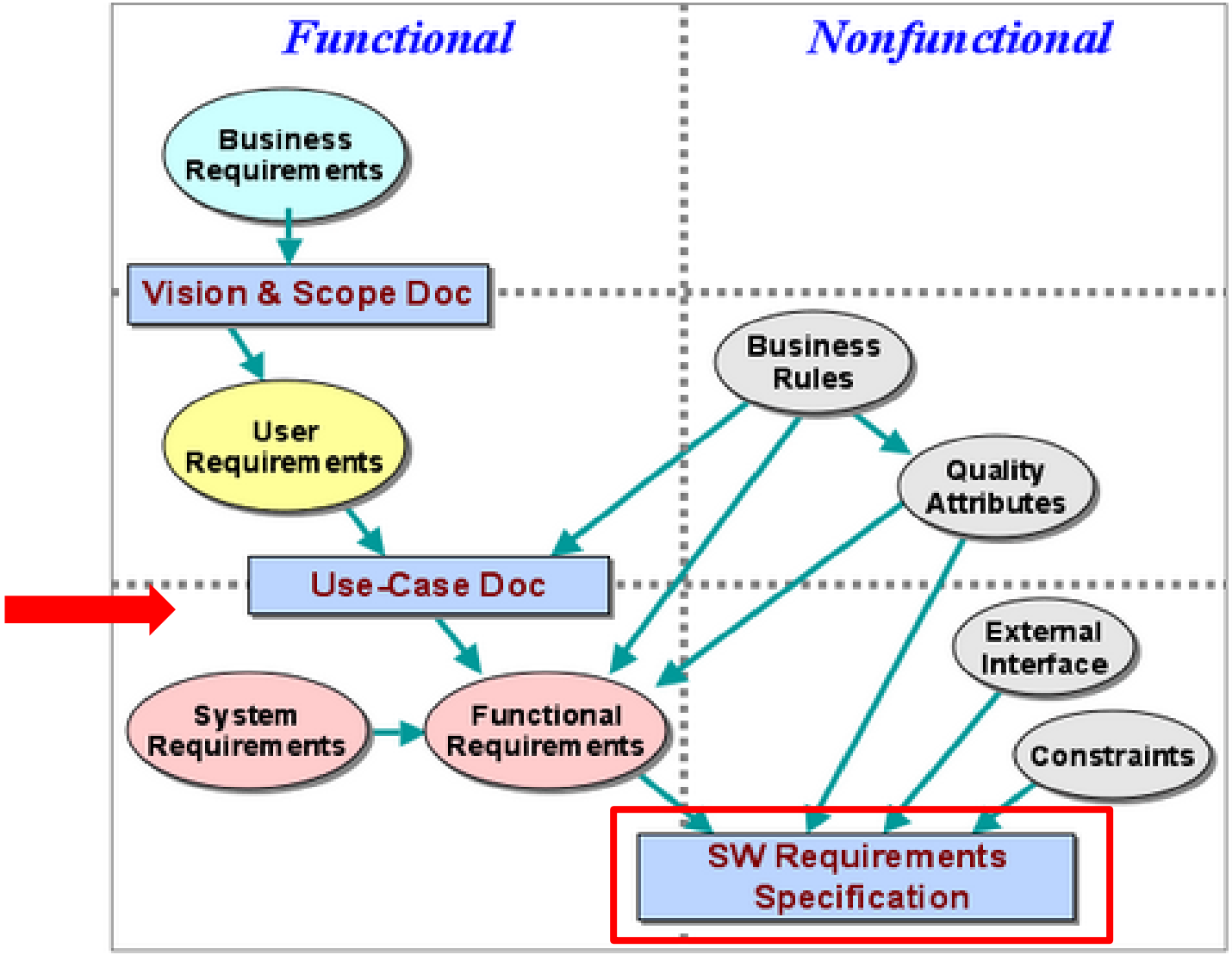


- Incremental, iterative, and interleaved

Writing Requirements



- **Use/Stakeholder requirements** (use case or story)
  - Descriptions of how the user will use, typically written in use case or user story
  - **Example**: "The registered users log into the system with ID and password."
- **System requirements**
  - Descriptions of system features
    - Functional requirements for system feature or functions, written in terms of input, processing, and output (or stimulus/response sequence)
    - Non-functional requirements for quality attributes such as scalability, efficiency
  - **Example**: "The system must allow the users to log into their account when they enter ID and password."
    - Requirement descriptions for developer
  - Use case or user story can be used for both user and system requirements.
- **Requirements analysis modeling for complex requirements**
  - Context diagram, activity diagram, data flow diagram, use case diagram


Functional
Nonfunctional
Business Requirements
Vision & Scope Doc
User Requirements
Use-Case Doc
System Requirements
Functional Requirements
Business Rules
Quality Attributes
External Interface
Constraints
SW Requirements Specification

Use Cases vs Functional Requirements

- **Use case** takes actor's viewpoint, shows external behavior and appearance.
- **Developers** implement the functions from system's viewpoint, with internal behavior, algorithms, storage, etc. to meet the **actor's viewpoint**.

Use Case

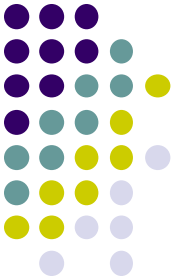
ID:	UC-6
Name:	Register for courses
Description:	Student accesses the system and views the courses currently available for him to register. Then he selects the courses and registers for them.
Primary Actor:	Student
Preconditions:	Student is logged into system
Postconditions:	Student is registered for courses
Main Success Scenario: (Normal Flow)	1. Student selects "Register New Courses" from the menu. 2. System displays list of courses available for registering. 3. Student selects one or more courses he wants to register for. 4. Student clicks "Submit" button. 5. System registers student for the selected courses and displays a confirmation message.

Action word (**verb**)

Actors' tasks

A sequence of interactions between actor and system to develop

Some Words to Avoid in writing requirements

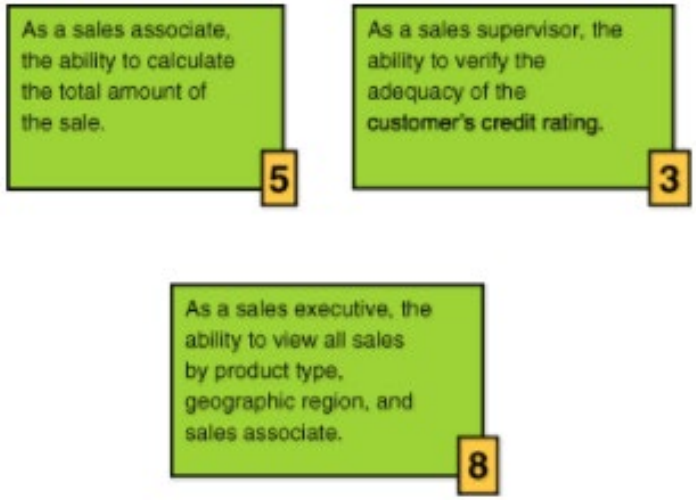


- Minimize, maximize, optimize
- User-friendly, rapid, easy, simple, intuitive, efficient, flexible, robust
- Seamless, transparent, graceful
- Improved, state-of-the-art, superior
- Sufficient, adequate, at least
- Reasonable, where appropriate, to the extent possible, if necessary
- Few, several, some, many
- Etc., including, and/or
- Optionally and many other adverbs
- Support
- Symbols or abbreviations, i.e. (that is), e.g. (for example)

User Story

- **User story** is a brief description of feature for the end user commonly used in **agile process**.
  - Simplified use case, typically written on a card
- Key elements of a user story
  - **Story id** and **name** (action word)
  - **User** (actor)
  - **Feature description** for the *user*, doing a *function* (task or steps to perform) to achieve a *goal*
    - Non-functional requirements can be specified as constraints

Informal story cards



Story card provides a mobile, tactile medium that team members can write on, shuffle about on the table.

Story Card Information

- Story identifier and name
- Story description: A sentence or two that describes the feature in customer terms
- Story type (C=customer domain, T=technology domain)
- Estimated work effort: The estimated work effort needed to deliver the story, including time for requirements gathering, design, coding, testing, and documentation
- Estimated Value Points (described in Chapter 8)
- Requirements uncertainty (erratic, fluctuating, routine, stable): An "exploration factor" for a specific story
- Story dependencies: Dependencies that could influence implementation sequencing
- Acceptance tests: Criteria the customer team will use to accept or reject the story

A well-structured story card

Story Card		Planned Iteration: 3	
Story ID: 25		Story Type:	Cust
Story Name: Establishment Sales Territories			
Story Description:			
As as Sales Manager, the ability to create U. S. sales territories based upon			
states and standard metropolitan areas.			
Est. Value Points:	13		
Est. Story Points:	8		
Requirements Uncertainty (E, F, R, S):	R (routine)		
Dependencies with other Stories:	None		
Acceptance Tests:			

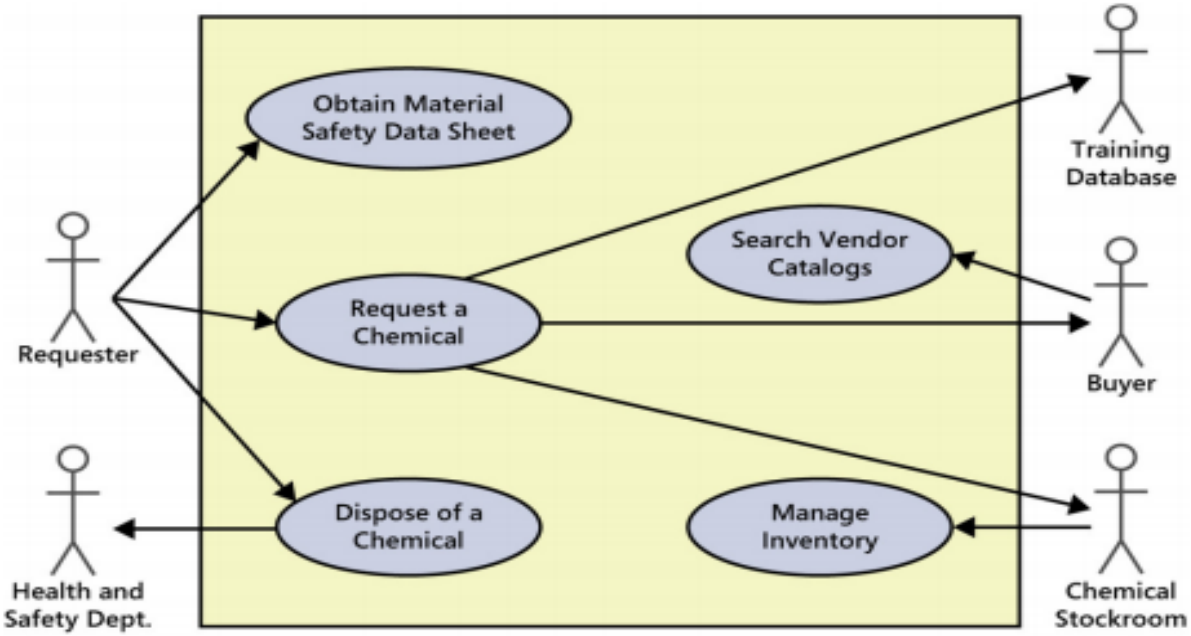
Pros and Cons of Use cases or stories

- Use cases works well for:
  - End-user applications
  - Websites
  - Devices that require user interactions
  - Services provided by one system to another

- Use cases aren't as valuable for:
  - Batch processes
  - Computation-intensive systems
  - Database or data processing projects

Requirements Analysis Modeling

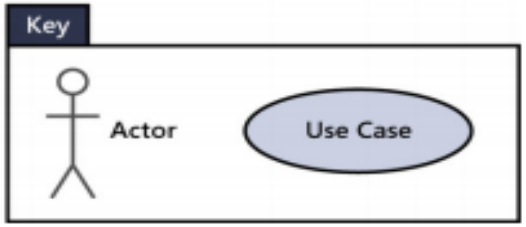
- Unified Modeling Language (UML) building blocks for **use case diagram**:
  - **Circle**: use case; **Human shape**: actor, **Box**: subsystem, system boundary; **Line**: association between actor and use case; Arrow: the direction of initiation, also indicating primary actor→ and →secondary actor.
  - **Use case names** are typically written: **Active verb** + **Object**



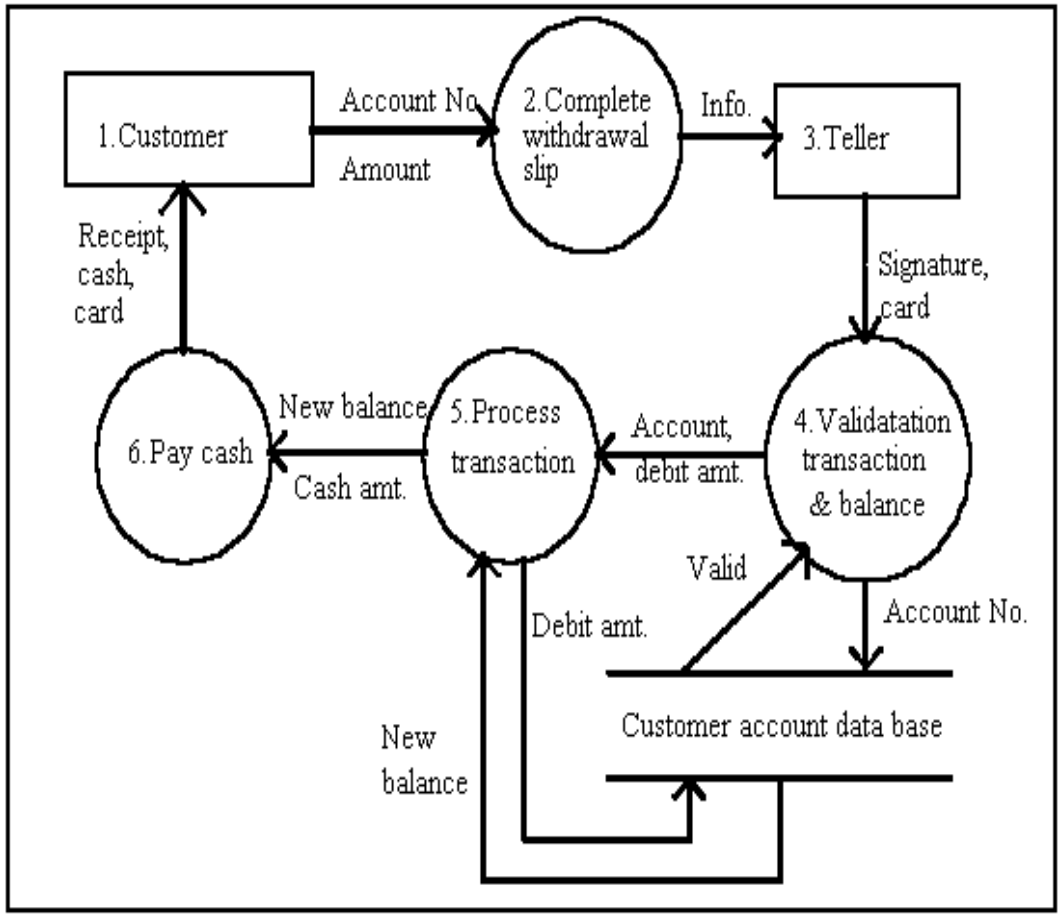
Training Database as an external system

Secondary actors typically on the right If the direction of initiation for a use is not clear.

Primary actors typically on the left if the direction of initiation for a use is not clear.



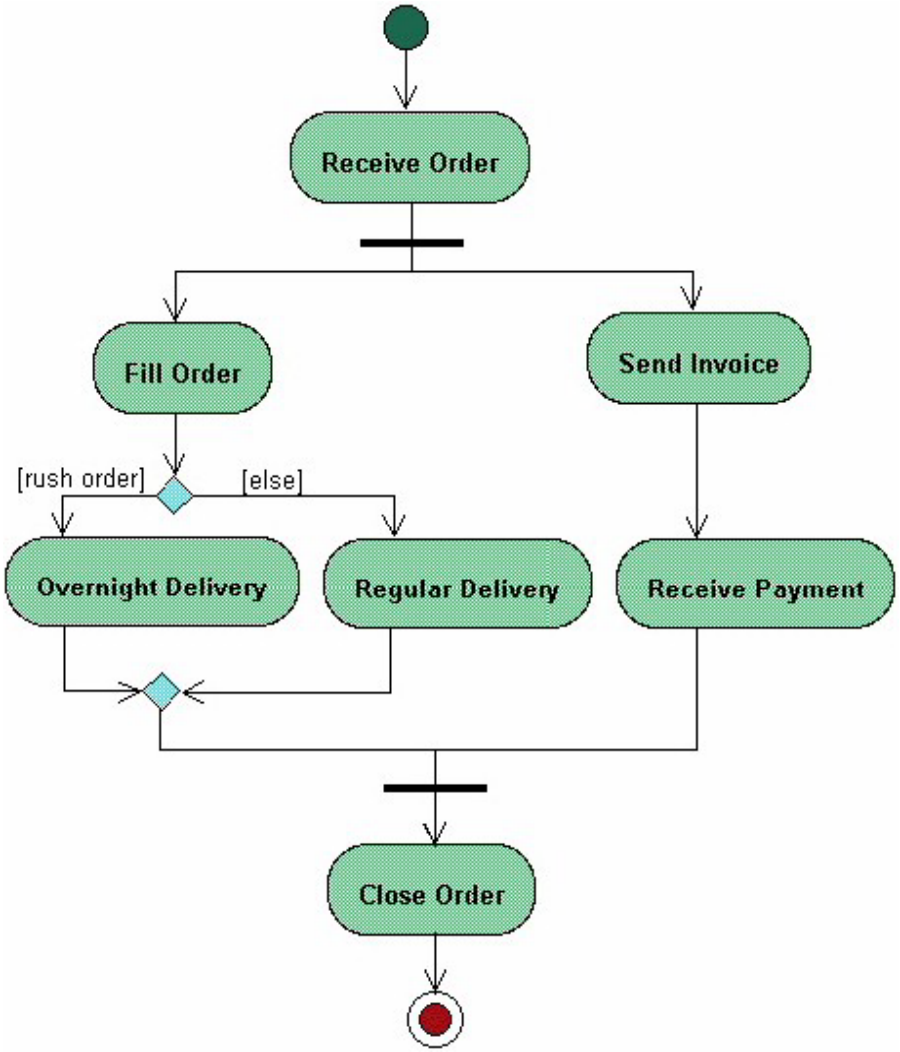
Requirements Analysis Model



R1: all processes have both input and output
R2: all processes are connected to either other processes entities or database
R3: all processes have unique names & numbering, e.g., sub-processes shall follow the numbering scheme of the parent process
R4: entities/DB may connect only to processes
R5: Information continuity must be kept. Information Continuity: The net I/O to the refinement must remain the same.
R6: The refinement per level should be 3 ~7.

- A **Data Flow Diagram** for cash withdrawal process

Requirements Analysis Model

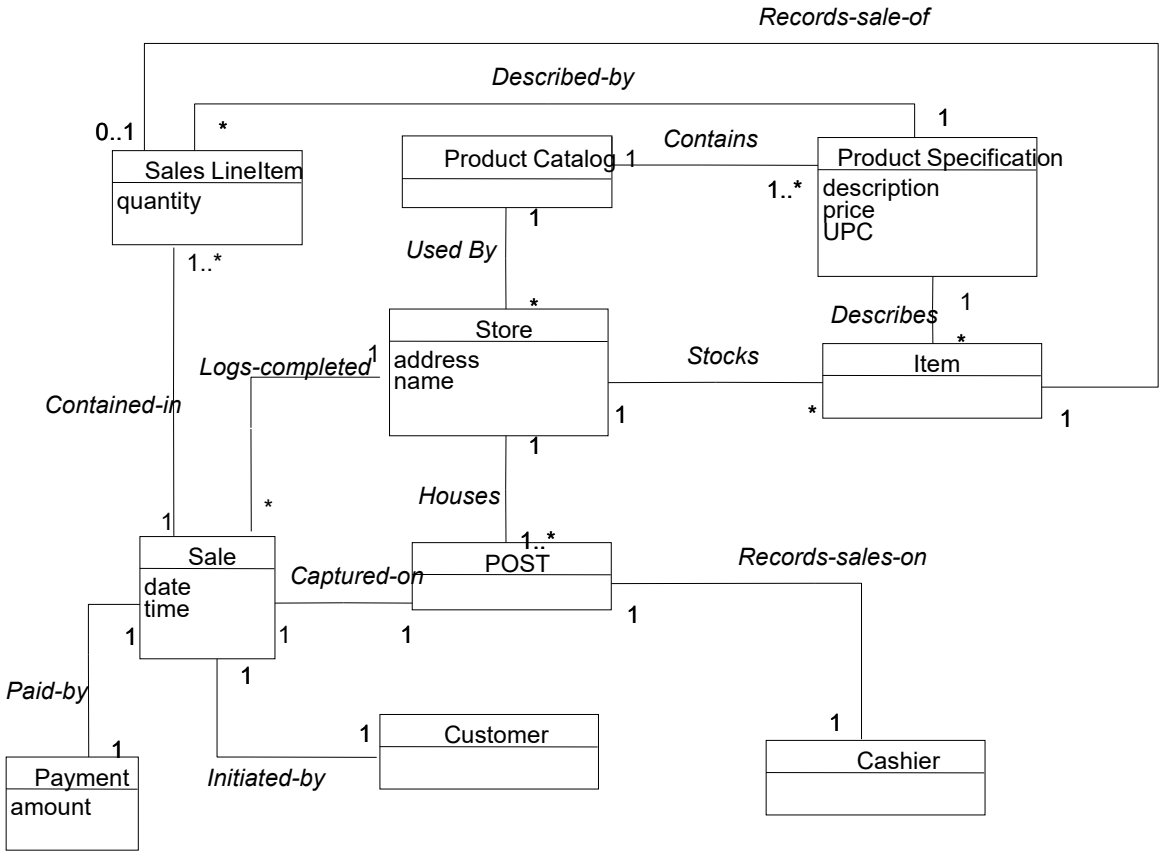


- An **Activity Diagram** for order workflow

+If necessary, supplements the use case by providing a graphical representation of the flow of interaction or business process within a specific scenario.

You create requirements analysis models only when necessary as it can be considered as non-productive activity.

Requirements Analysis Model

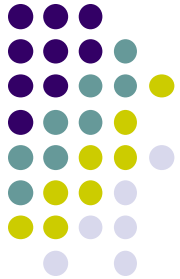


A **domain model** for Use Case "Buy Items"

Domain modeling **reduces** the representation **gap** between mental model on business and application and software object model.

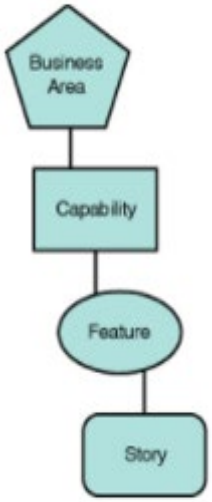
For example, a Sale in the domain model is a concept but a Sale in the design model is a software class.

Requirements Modeling for Web Applications (requires additional activities)



- **Content analysis**
  - The full spectrum of content to be provided by the WebApp, including text, graphics and images, video, and audio data
- **Interaction analysis**
  - The manner in which the user interacts with the WebApp
    - Use-cases and navigational modeling can be developed to provide detailed descriptions of this interaction and a flow of interaction.
- **Functional analysis**
  - All operations and functions
    - An activity diagram can be used to represent processing flow.
    - The use-cases created as part of interaction analysis define the operations and imply other processing functions.
- **Configuration analysis**
  - The environment and infrastructure (both server and client sides) in which the WebApp resides

A Hierarchy of Product Requirements



(Each level in is about **3 times** the lower one.)

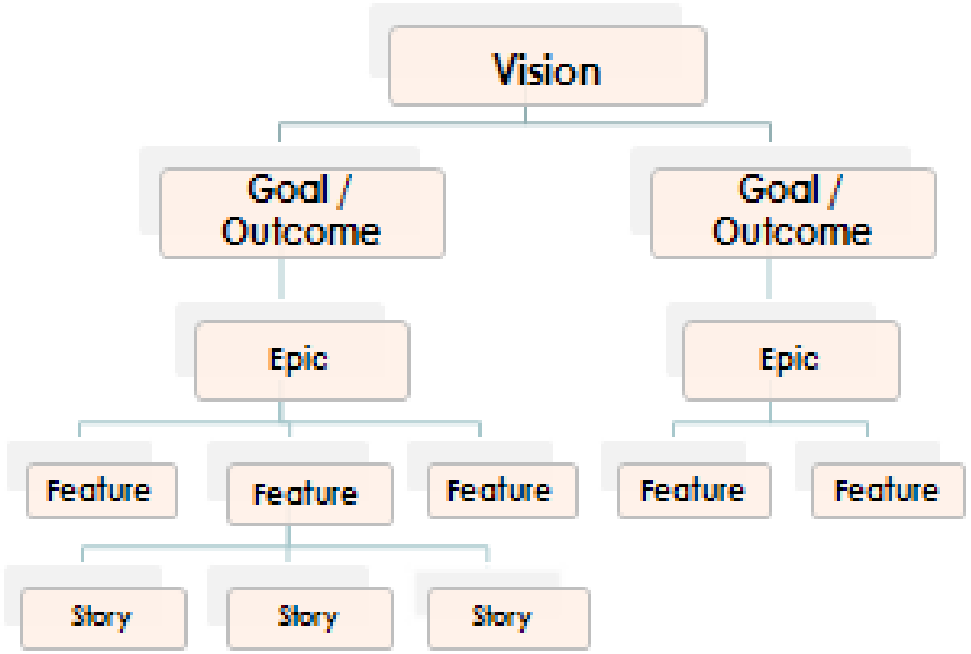
- **Business area**
- **Capability** (epic, big use case, or business function)
  - High-level business or product function that is complete and valuable
    - May require multiple iterations taking 20 –100 days of work
    - The **DoD** uses "capability-based planning" with flexibility and predictability
- **Feature**
  - A piece of a product function that delivers some useful and valuable functionality, e.g., a feature for customer credit checking
    - May take 1-2 iterations taking 6 – 30 days
- **Story**
  - A piece of a feature that is useful functionality, but may not be a complete function to deliver, e.g., the customer credit checking needs several stories
    - May take one iteration or less taking 2 – 10 days
    - Sometimes the difference between feature and story can be vague.

Feature-Story Example

Feature: As a credit analyst I need the ability to check a customer's credit rating.

- Story 1: As a credit analyst I need the ability to check the prior payment history with this customer.
- Story 2: As a credit analyst I need the ability to check this customer's credit bureau status.
- Story 3: As a credit analyst I need the ability to calculate our internal credit rating based on history and credit report.

Feature Breakdown Structure (FBS): WBS based on Features

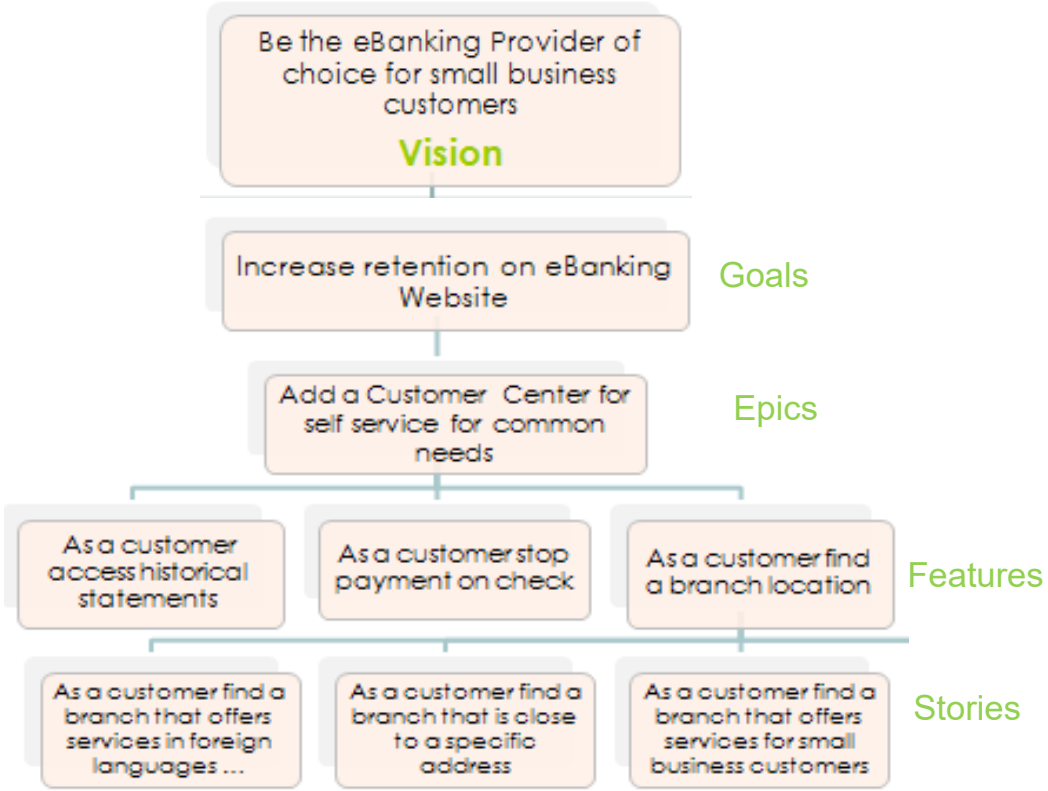


Product features or requirements breakdown by features (commonly used in Agile process)

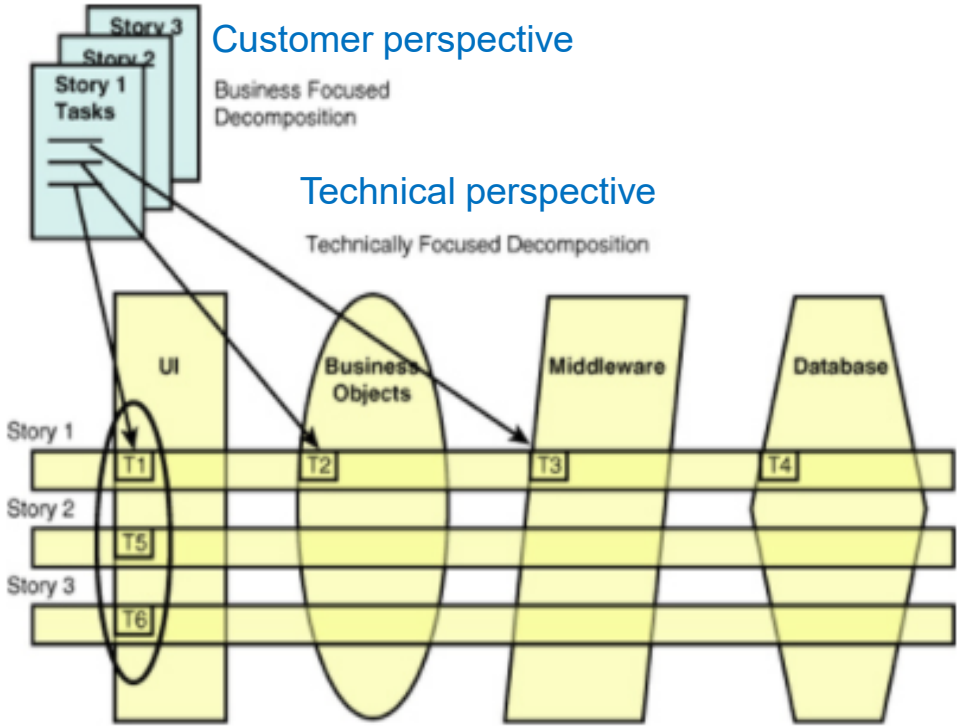
FBS can depict a product architecture.

- The **structure of requirements** aligned with **vision and goals**
  - **Vision** is a high-level statement about the product or business objectives.
  - **Project goals/outcomes** are to address the product vision.
  - **Epics** are large grained chunks of business value (several iterations to complete).
  - **Features** are smaller than epics, typically take 1-2 iterations.
  - **Stories** are user stories at the smallest level of requirements.

A Sample FBS


Be the eBanking Provider of choice for small business customers
Vision
Increase retention on eBanking Website
Goals
Add a Customer Center for self service for common needs
Epics
As a customer access historical statements
As a customer stop payment on check
As a customer find a branch location
Features
As a customer find a branch that offers services in foreign languages ...
As a customer find a branch that is close to a specific address
As a customer find a branch that offers services for small business customers
Stories

User Stories to Technical Activities



- **The development team**
  - turns the story cards to a list of technical activities required to implement the stories
  - uses the task breakdown to estimate the effort and risk from the stories