SOFT 423
Software Requirements

Section IV-1
Requirements Analysis

Outline

- Understanding User Requirements
- Understanding Business Rules
- Specifying Data Requirements

Understanding User Requirements

- Understand what the users intend to do
  - A product-centric approach: Defining the features to implement in the software
  - A user-centric and usage-centric approach: Focusing on users and their anticipated usage

Use Cases

- Use cases: Shift from product-centric approach to usage-centric elicitation
- The tasks of users performed with the system
- User-system interactions bring a valuable outcome for some stakeholders.
  - Derive the necessary functionality implemented to enable usage scenarios
  - Understand the user's requirements on many classes of projects

Use Cases

- Define a sequence of interactions between a system and an external actor
- The interactions enable the actor to achieve some outcome of value.
- Actors:
  - A role played by a person
  - Something or someone which exist outside the system.
  - Actors may be end users, other systems, or hardware devices
  - Take part in a sequence of activities with the system to achieve goals

Use Cases

- The names of use cases: the form of a verb followed by an object.
- Chemical Tracking System
  - Request a Chemical
  - Change a Chemical Request
  - Check Status of an Order
  - Generate Chemical-Usage Reports

Use Cases Examples

- Airport Check-in Kiosk
  - Check in for a Flight
  - Print Boarding Passes
  - Change Seats
  - Check Luggage
  - Purchase an Upgrade
- Online Bookstore
  - Update Customer Profile
  - Search for an Item
  - Buy an Item
  - Track a Shipped Package
  - Cancel an Unshipped Order

Use Cases

- Business Use Cases:
  - The processes of a business and their interactions with external parties like customers and partners.
  - The business process as a black box
  - A restaurant

    Actors: Waiter, Client, Cashier, Chief;
    Business Use Cases:
    Order Food, Serve Food, Eat Food, Pay for Food,
    Cook Food, Serve Wine, Drink Wine, Pay for Wine

Use Cases

- System Use Case:
  - Describe what the actor achieves interacting with the system
  - System functionality
  - A way in which a user of a system can make use of the system to get the result they want.
  - Bank ATM

Actors: Customer, ATM Technician, Bank;
System Use Cases:
Check Balances, Deposit Funds, Withdraw Cash, Transfer Funds, Maintenance, Repair

User Story

- User Story:
  - Short, simple description of a feature told from a user of the system
  - Template

    As a <type of user>, I want <some goal> so that <some reason>.

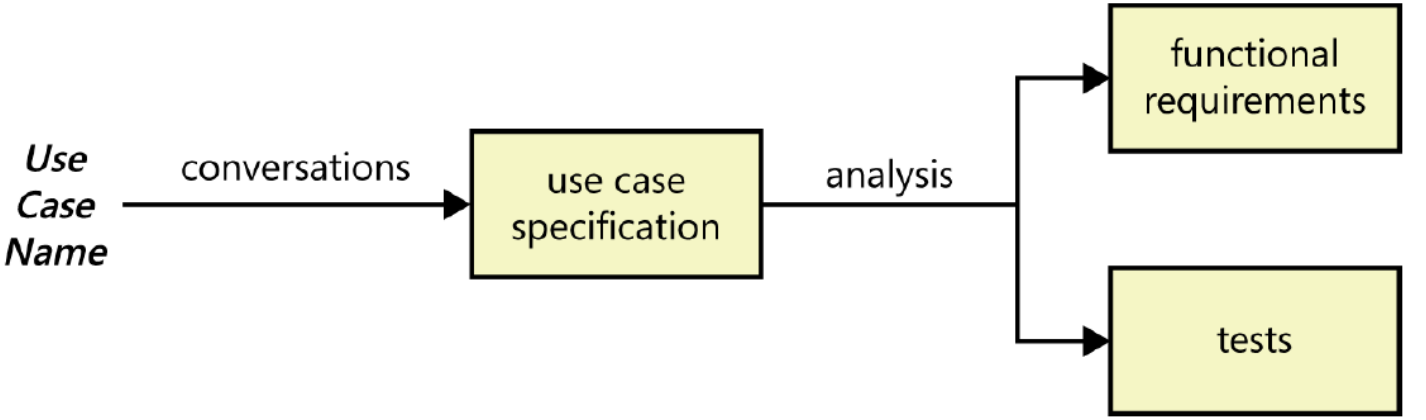
Application	Sample use case	Corresponding user story
Chemical tracking system	Request a Chemical	As a chemist, I want to request a chemical so that I can perform experiments.
Airport check-in kiosk	Check in for a Flight	As a traveler, I want to check in for a flight so that I can fly to my destination.
Accounting system	Create an Invoice	As a small business owner, I want to create an invoice so that I can bill a customer.
Online bookstore	Update Customer Profile	As a customer, I want to update my customer profile so that future purchases are billed to a new credit card number.

Use Cases

- Simple technique which helps
  - identify tasks
  - identify actors and how they interact
- A sequence of actions with an observable result of value to a particular actor

Use Cases

- Develop the function requirements
- Develop the testing metrics.



- Use case brief format
- Use case long format
- Use case diagram
  - Relationships of actors and use cases
  - Associations of Use Cases

Use Case Brief format example

- Place Skype Call:
  - The user is logged into skype. They select a contact from their contact list and select the place call button. A separate window appears that manages the call. At the end of the call, the user presses the end call button which returns the user to the contact list

Use Case Long format example

1. Brief Description

This use case describes how a skype user places an audio only skype phone call

2. Actors

2.1 User

2.2 Callee

3. Preconditions

The user has an active internet connection

The user is logged into skype

Use Case Long format example

4. Basic flow of events

1. The user selects an online contact
2. The system expands the contact to include an instant message window, and several buttons including the place call button.
3. The user presses the place call button
4. A window appears with the message calling
5. The message changes to ringing
6. The callee answers the call
7. Message connected, audio is transmitted both ways
8. The user presses the end call
9. The window disappears returning the user to the contact list

Use Case Long format example

5. Alternative Flows

5.1 Callee refuses call

1. In step 6, the callee refuses the call
2. The message changes to caller refused call
3. User closes the window returning to contact list

5.2 Callee ignores call

1. In step 6, callee ignores calls
2. Message ringing continues
3. User presses end call
4. The window disappears returning the user to the contact list

Use Case Long format example

5.3 User cancels call

1. In step 4 or 5, user decides not to call
2. User presses cancel call button
3. The window disappears returning the user to the contact list

6 Exceptions

1. Message - connection lost display
2. User closes message window
3. Contact lists are updated to offline.

7 Post-conditions

User is returned to contact list

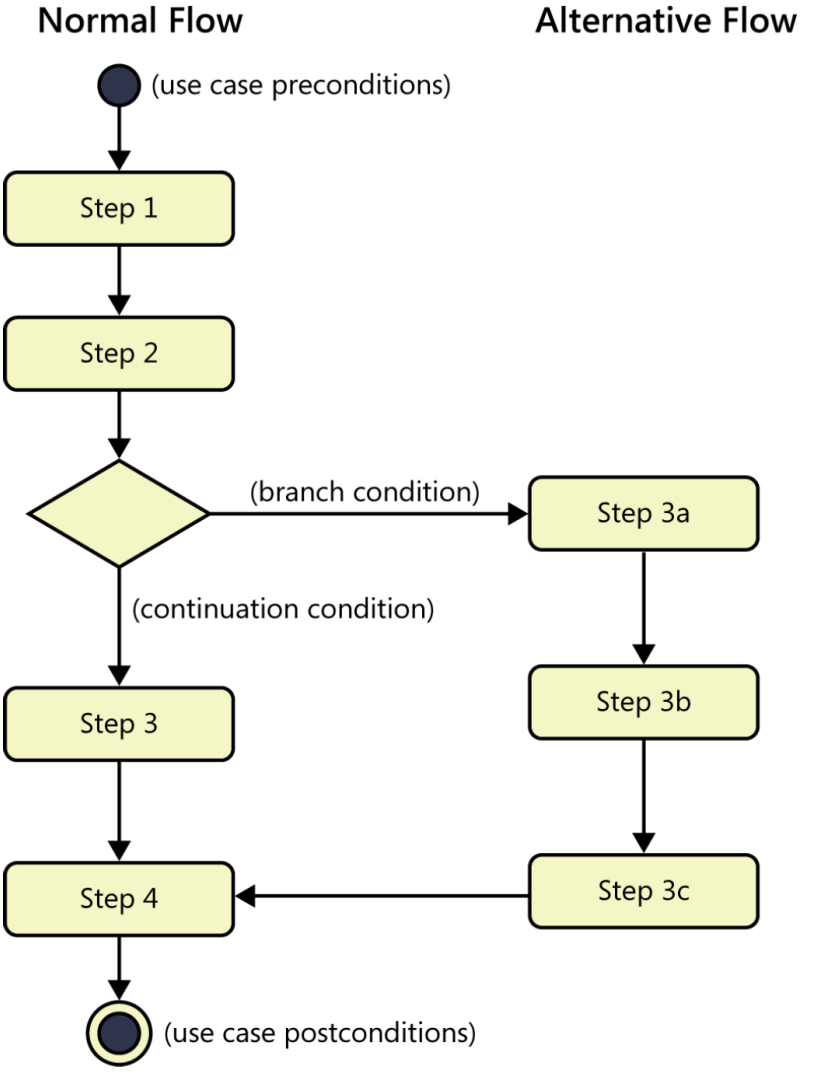
Use Cases - Anatomy

- ID and Name
  - A unique identifier and a succinct name
- Brief Description
  - Purpose in a few sentences
- Actor(s)
  - Identify the actors involved
- Flow of Events
  - Usually a textual description
  - Normal flow, alternative flows and exceptions
    - Normal flow: The basic instance of the use case
    - Alternative flows: Other successful instances
    - Exceptions: Conditions that have the potential to prevent a use case from succeeding.

Use Cases - Anatomy

A flowchart /
An activity diagram

visually represent the logic flow in a complex use case



Use Cases - Anatomy (optional)

- Preconditions
  - The prerequisites that must be met before the use case can be executed.
  - *The customer can withdraw money with the precondition that there are money in ATM.*
  - *The user's identity has been authenticated.*

Use Cases - Anatomy (optional)

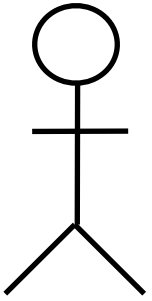
- Postconditions
  - State of system after the use case is executed
    - Something observable to the user (*The system displayed an account balance*).
    - Physical outcomes (*The ATM has dispensed money and printed a receipt*).
    - Internal system state changes (*The account has been debited by the amount of a cash withdrawal, plus any transaction fees*).

Review

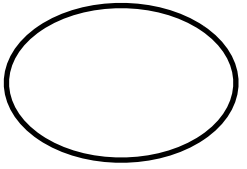
- Business Use Case and System Use Case
- Use Case Brief Format
- Use Case Long Format
  - Normal Flow, Alternative Flow and Exception
  - Precondition and Postcondition

Use Case diagrams

- **Behavior diagrams** used to describe a set of use cases that the system can perform in collaboration with the actors.
  - context of the use case with other use cases
  - relationship between actors and use cases



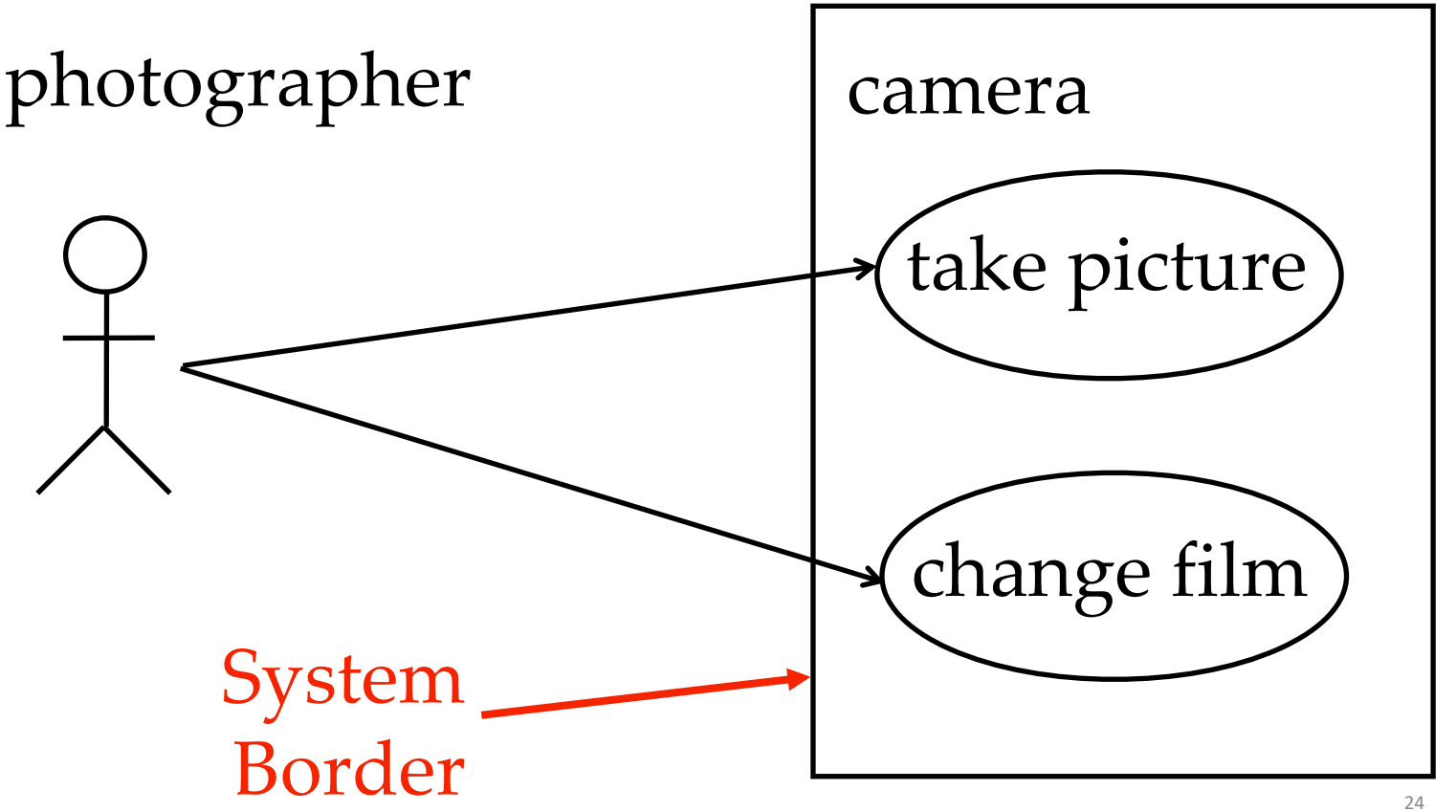
Actor - use a box if actor is not human



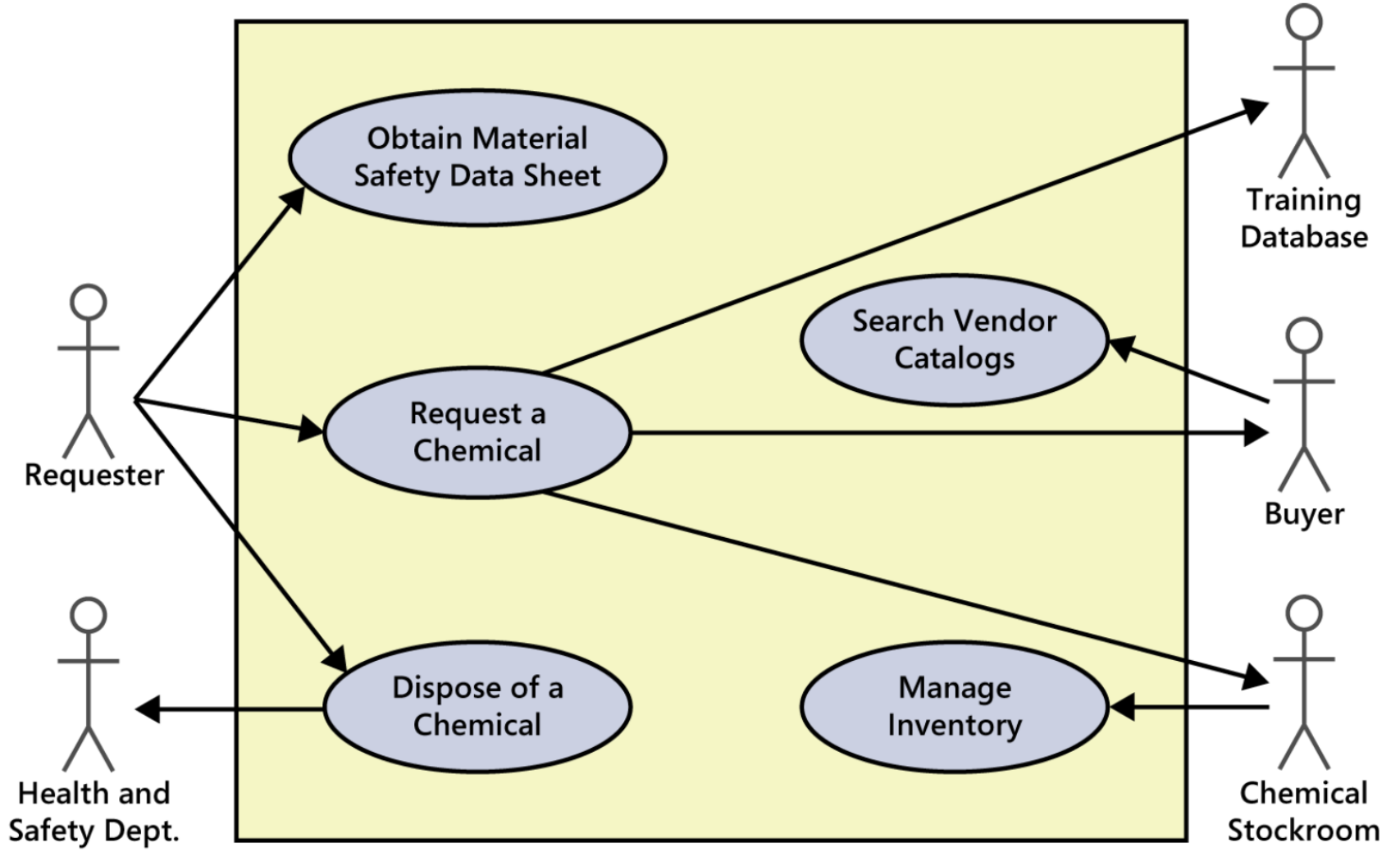
Use Case

Use Case Diagram example

An association is the relationship between an actor and a use case.



Use Case Diagram



Use Case Diagram example

Arrows from each actor connect to the use cases with which the actor interacts.

Primary actor: An arrow from a primary actor to a use case.

Secondary actor: An arrow goes from a use case to a secondary actor, who participates somehow in the successful execution of the use case.

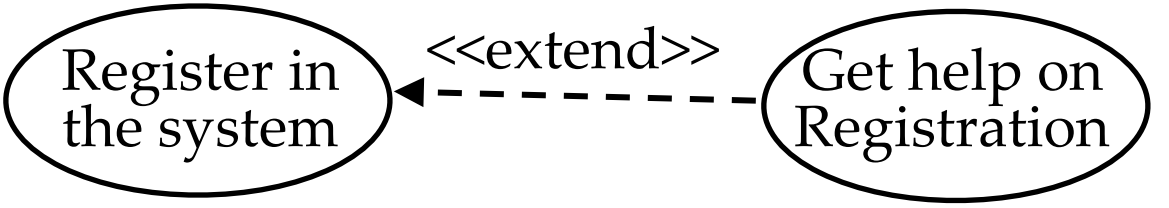
The primary actor initiates the use case and derives the main value from it.

Other software systems often serve as secondary actors, contributing behind the scenes to the use case execution.

Use Case Diagram-Extend

<<Extend>>

A directed relationship that specifies how and when the behavior defined in **the extending use case** can be inserted into the behavior defined in **the extended (base) use case**.



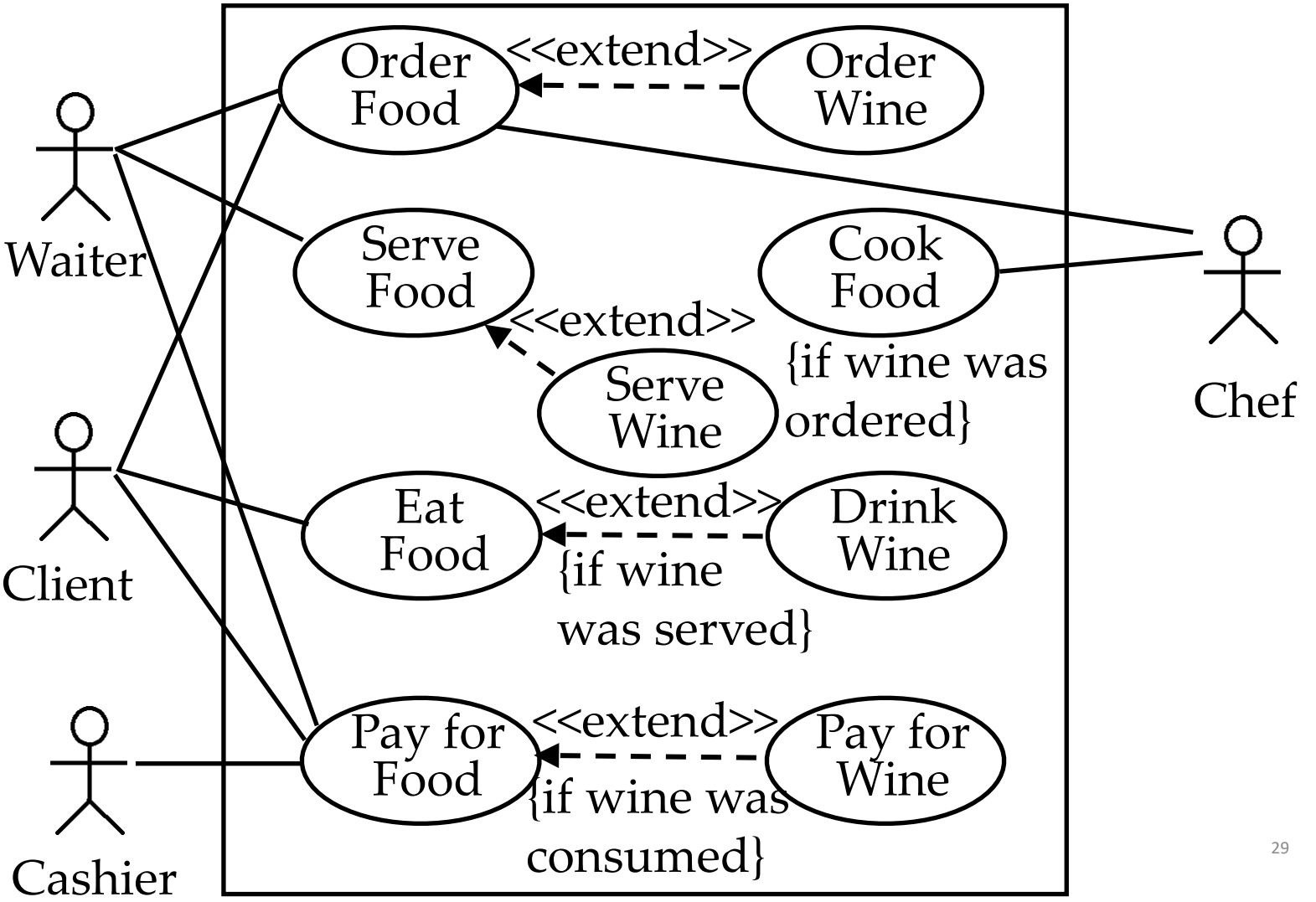
A dashed Line with arrow: From extending use case to the extended use case, with the label <<extend>>.

Use Case Diagram-Extend

<<Extend>>

- **Extended use case** is meaningful on its own, and independent
- **Extending use case** defines optional behavior
- **The extend relationship** is owned by the extending use case
- **The extension** takes place at one or more extension points defined in the extended use case
- A standalone use case extends the normal flow into an alternative flow.

Use Case Diagram-Extend



Use Case Diagram-Include

<<include>>

A directed relationship between two use cases which shows that behavior of **the included use case** is inserted into the behavior of **the including (the base) use case**.

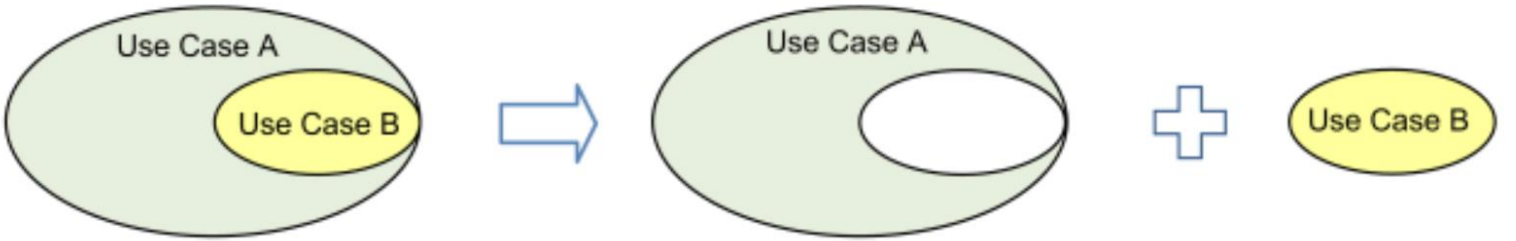
The include relationship could be used:

- to simplify a large use case by splitting it into several use cases;
- to extract **common parts** of the behaviors of two or more use cases.

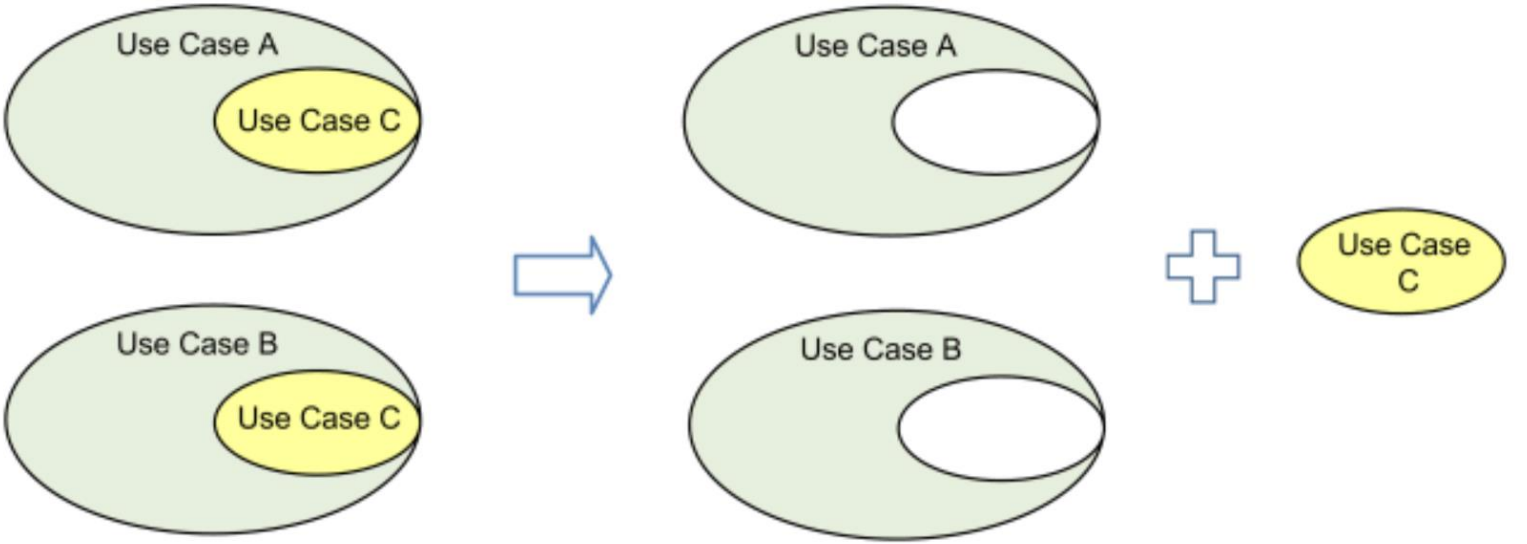
Use Case Diagram-Include

<<include>>

B is extracted from larger A into a separate use case



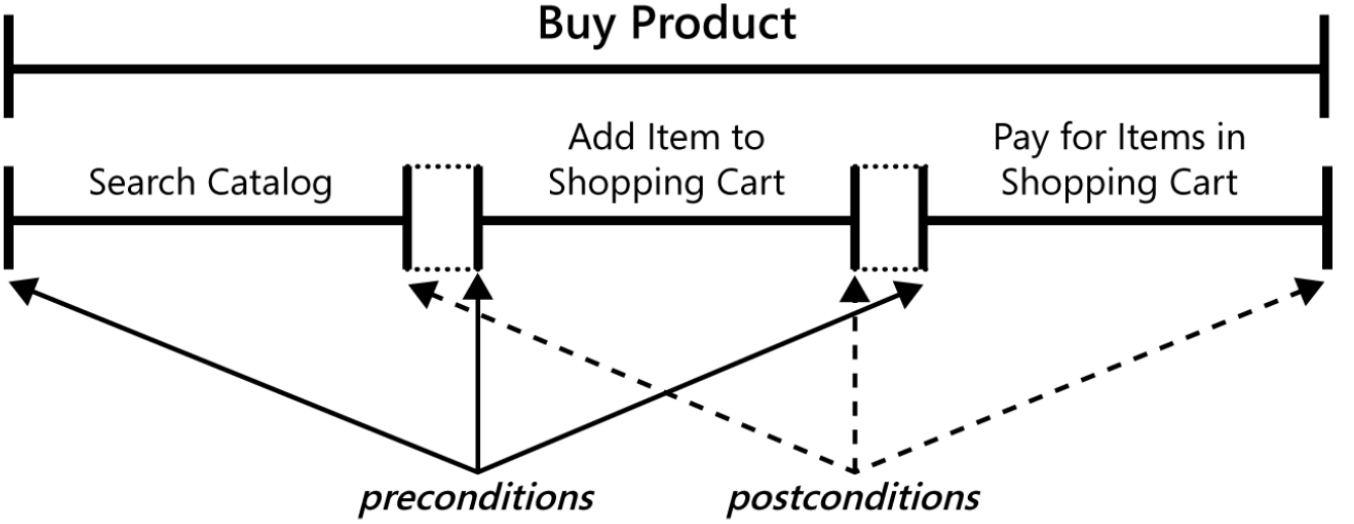
C is extracted from A and B to be reused by both use cases



Use Case Diagram-Include

<<include>>

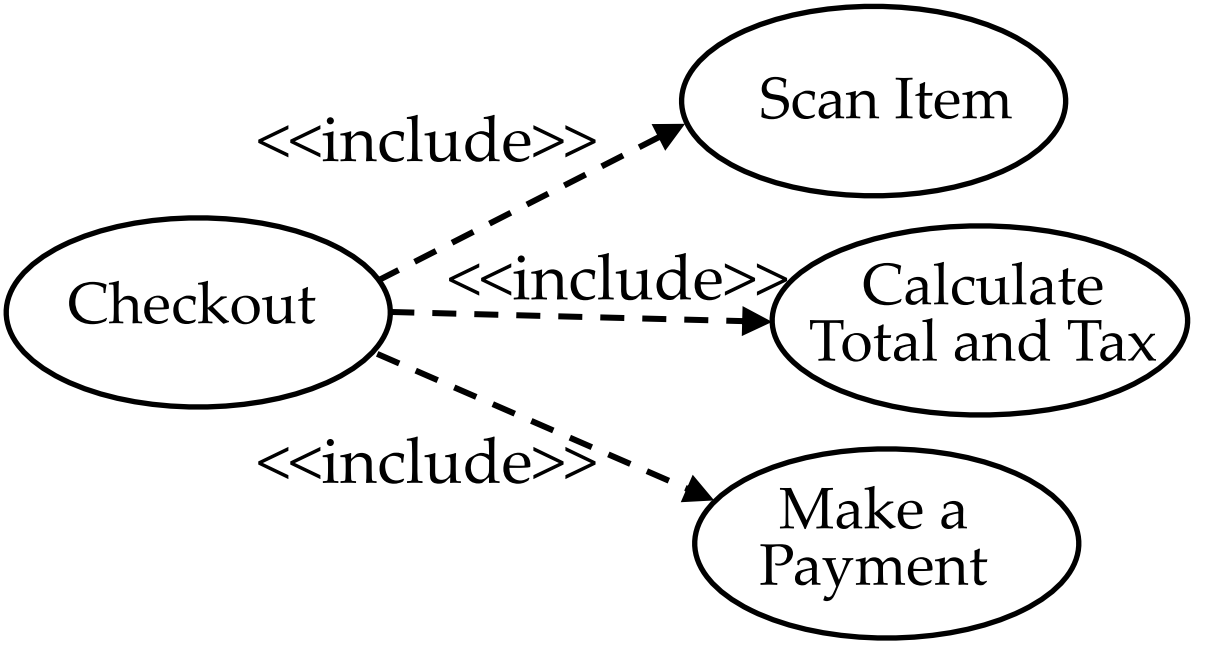
- The included use cases can be independently performed.
- The included use cases are performed in sequence as a single large use case.



Use Case Diagram-Include

<<include>>

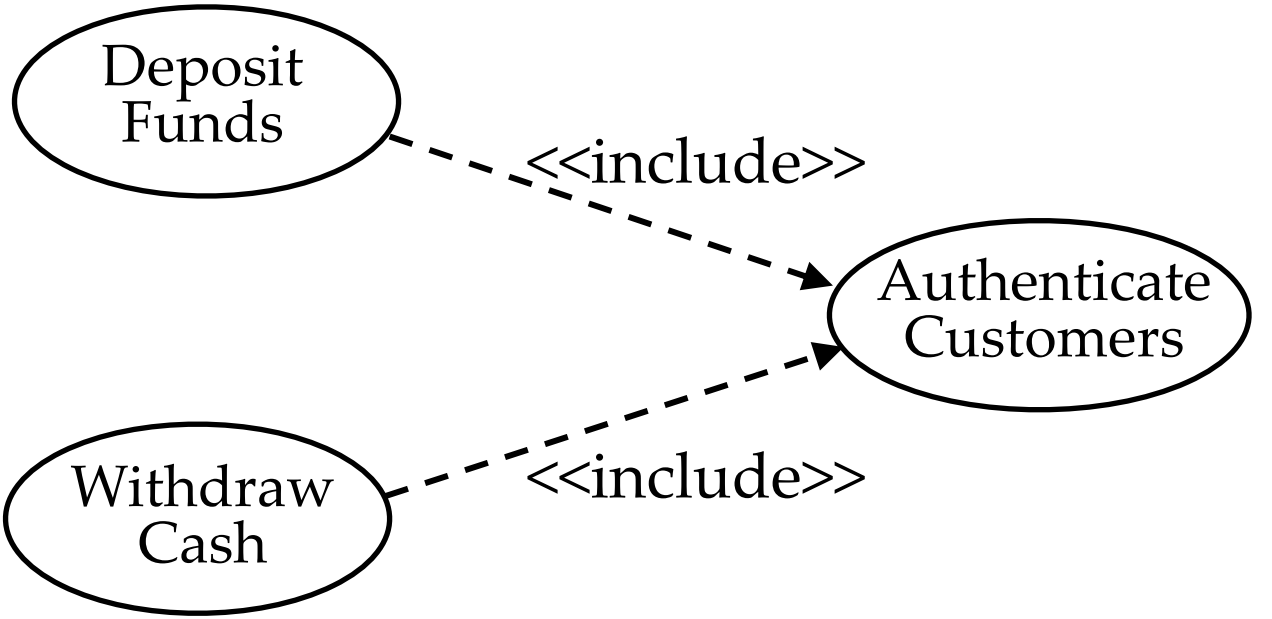
A dashed arrow with an open arrowhead from the including (base) use case to the included use case.
Checkout use case includes several use cases - Scan Item, Calculate Total and Tax, and Payment



Use Case Diagram-Include

<<include>>

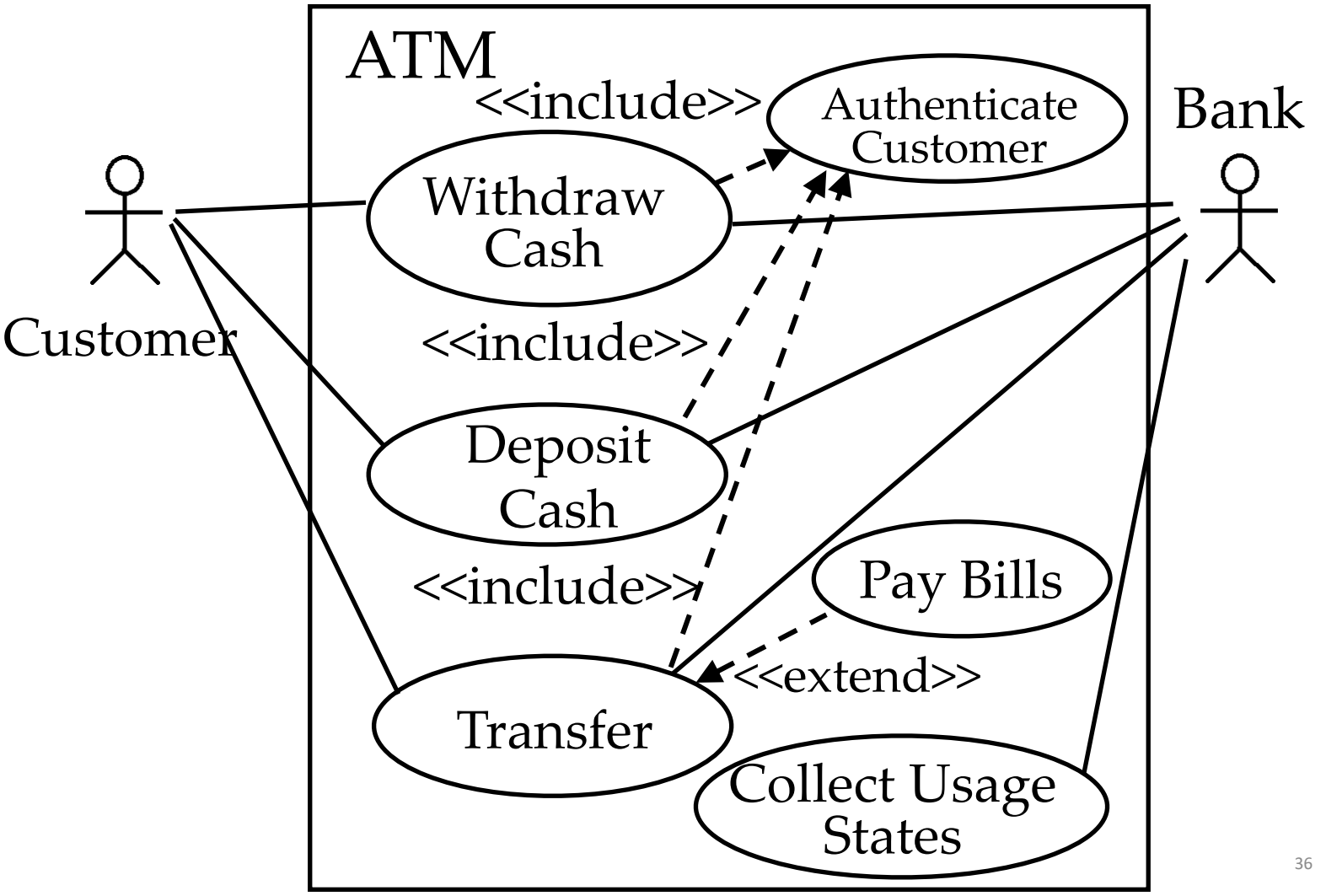
Deposit Funds and Withdraw Cash use cases include Authenticate Customers use case.



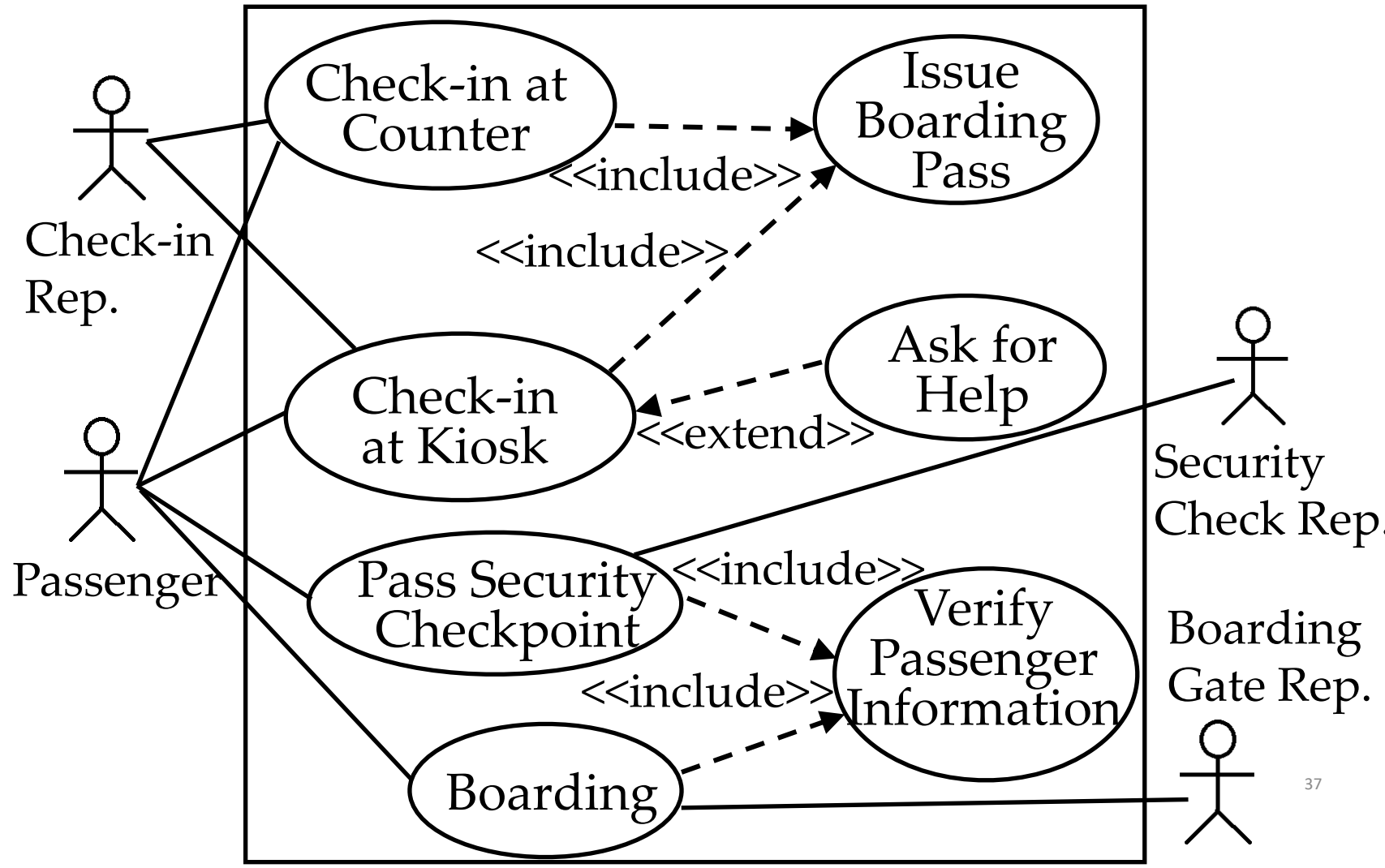
Review

- Use Case Diagrams
  - context of the use case with other use cases
  - relationship between actors and use cases
    - Extend
    - Include

Use Case Diagram-ATM



Use Case Diagram-Airport



Building Use Cases

- Identify and Describe the Actors
  - who uses the system?
  - who gets information from the system?
  - where in the company is the system used?
  - who supports and maintains the systems?
  - what other systems use the system?

Building Use Cases

- Identify Use Cases and Write Brief Descriptions
  - what will the actor use the system for?
  - will the actor create, add, remove, access 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 occurrences in the system

Building Use Cases

- Identify Actor–Use Case Relationships
  - some use cases involve multiple actors
  - often represented pictorially
- Outline the Individual Use Cases
  - outline basic flow first
    - what event starts, how does it end, does it repeat?
  - add alternative flows
    - optional situations?
    - what might go wrong? what might not happen? resources?

Building Use Cases

- Refine the use cases
  - alternate flows
  - pre/post conditions
- Discover exceptions by considering preconditions are not reached.
- Write the long description of use cases

Use Case Benefits

- Relatively easy to write
- Focus on users
- Easy for users to understand, engage users
- Useful later for design and testing
- Help with documentation
- Useful for requirements prioritization

Outline

- Understanding User Requirements
- Understanding Business Rules
- Specifying Data Requirements

Understanding Business Rules

- Business Rules
  An extensive set of policies, laws, standards, and government regulations
- Software applications need to enforce business rules.
  "Contoso requires an annual refresher class in the safe handling of hazardous chemicals."
  "No credit check is to be performed on return customers when they are applying for visa."

Understanding Business Rules

- A property of the business, not software requirements.
- A rich source of requirements because they dictate properties the system must possess to conform to the rules.
- Business rules influence business requirements, user requirements, functional requirements, and quality attributes.

Business Rules Influence-Examples

Influence on Business Requirement:

- Government regulations can lead to necessary business objectives for a project.

The Chemical Tracking System must enable compliance with all federal and state chemical usage and disposal reporting regulations within 5 months.

Business Rules Influence-Examples

Influence on User Requirement:

- Privacy policies dictate which users can and cannot perform certain tasks with the system.

Only laboratory managers are allowed to generate chemical exposure reports for anyone other than themselves.

Business Rules Influence-Examples

Influence on Functional Requirement:

- Company policy is that all vendors must be registered and approved before an invoice will be paid.

If an invoice is received from an unregistered vendor, the Supplier System shall email the vendor editable PDF versions of the supplier intake form and the W-9 form.

Business Rules Influence-Examples

Influence on Quality Attribute:

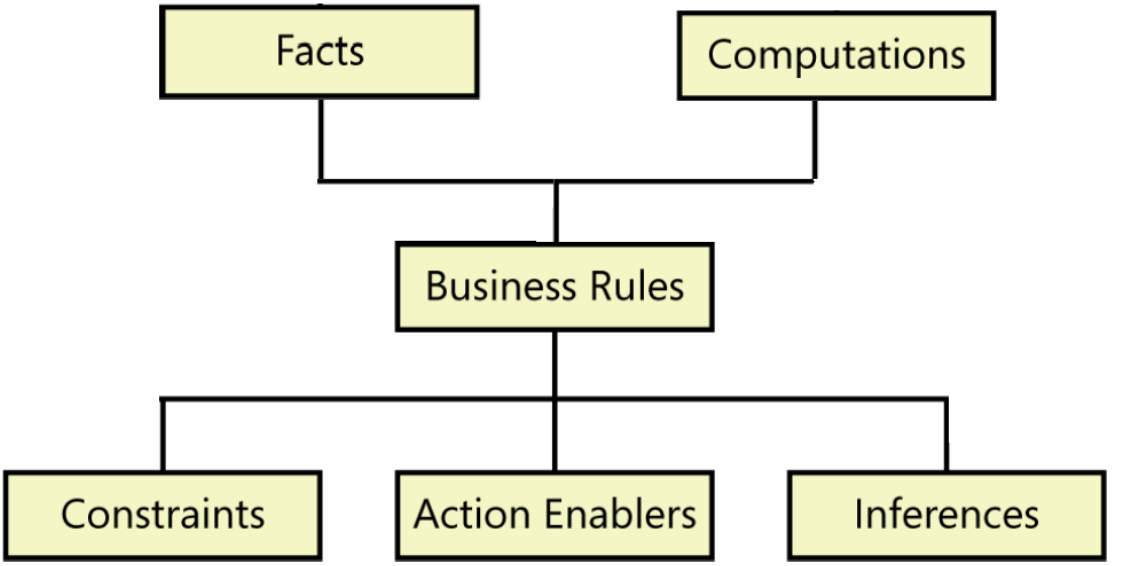
- Regulations from government agencies, such as OSHA and EPA, can dictate safety requirements, which must be enforced through system functionality.

The system must maintain safety training records, which it must check to ensure that users are properly trained before they can request a hazardous chemical.

Review

- Building Use Cases
  - Identify Actors
  - Identify Use Cases
  - Identify Act-Use Case Relationships
- Use Case Benefits
- Business Rules
  - Definition
  - Business Rules' Influence

Business Rules Taxonomy



- Classifying the rules gives you an idea of how you might apply them in a software application.

Business Rules - Facts

- A fact describes associations or relationships between important business terms.
- Focus on facts that are in scope of the project
  - *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.*
- A statement that is true about the business at a specified point in time.

Business Rules - Constraints

- A statement that restricts the actions that the system or its users are allowed to perform.
- Certain actions *must* or *must not* or *may not* be performed, or that *only* certain people or roles can perform particular actions.
- Software projects have many kinds of constraints.
  - Project level constraints: Schedule, staff, and budget limitations.
  - For developers: Product design and implementation constraints.
  - Business rules: Constraints on the way the business operates.

Business Rules - Constraints

- **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.

- **Government regulations**

Airline pilots must receive at least 8 continuous hours of rest in every 24-hour period.

- **Industry standards**

Mortgage loan applicants must satisfy the Federal Housing Authority qualification standards.

Business Rules - Constraints

Roles and Permissions Matrix	Employee	Administrator	Circulation Staff	Library Aide	Non-Employee	Volunteer	Patron
System Operations							
Log in to library system		X	X	X			
Set up new staff members		X					
Print hold pick list		X	X	X			
Patron Records							
View a patron record		X	X				
Edit a patron record		X	X				
View your own patron record		X	X	X		X	X
Issue a library card		X	X				
Accept a fine payment		X	X				
Item Operations							
Search the library catalog		X	X	X		X	X
Check out an item		X	X				
Check in an item		X	X	X		X	
Route an item to another branch		X	X	X		X	

Business Rules – Action Enablers

- A rule that triggers some activity if specific conditions are true.
- A rule might lead to specifying software functionality that makes an application exhibit the correct behavior when the system detects the triggering event.
- A decision table provides a concise way to document action-enabling business rules that involve extensive logic.

Business Rules – Action Enablers

A decision table to define whether a chemical request is accepted or not under the conditions.

Requirement Number					
Condition	**1**	**2**	**3**	**4**	**5**
User is authorized	F	T	T	T	T
Chemical is available	—	F	T	T	T
Chemical is hazardous	—	—	F	T	T
Requester is trained	—	—	—	F	T
Action					
Accept request			X		X
Reject request	X	X		X	

Business Rules – Action Enablers

- The format of an enabler: "If <some condition is true or some event takes place>, then <something happens>

  If the chemical stockroom has containers of a requested chemical in stock, then offer existing containers to the requester.

- Businesses develop policies that are intended to enhance their commercial success

  If the customer ordered a book by an author who has written multiple books, then offer the author's other books to the customer before completing the order.

Business Rules – Inferences

- Inferred Knowledge / Derived Fact
- An inference creates a new fact from other facts.
- "if/then" pattern: "then" clause of an inference simply provides a piece of knowledge, **not an action to be taken**.

  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.

Business Rules – Computations

- Transform existing data into new data by using specific mathematical formulas or algorithms.
- Computations follow rules that are external to the enterprise, such as income tax withholding formulas.

  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 total price for an order is the sum of the price of the items ordered, less any volume discounts, plus state and county sales taxes for the location to which the order is being shipped, plus the shipping charge, plus an optional insurance charge.

Business Rules – Computations

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.

ID	Number of units purchased	Percent discount
DISC-1	1 through 5	0
DISC-2	6 through 10	10
DISC-3	11 through 20	20
DISC-4	More than 20	30

- Boundary value overlaps

Atomic Business Rules

- Write the business rules at the atomic level.
- Atomic: The business rules can't be decomposed.
- Keeps your rules short and simple.
- Don't use "or" logic on the left-hand side of an "if/then", and avoid "and" logic on the right-hand side.

"You can check out a DVD or Blu-ray Disc for one week, and you may renew it up to two times for three days each, but only if another patron hasn't placed a hold on it."

Atomic Business Rules

"You can check out a DVD or Blu-ray Disc for one week, and you may renew it up to two times for three days each, but only if another patron hasn't placed a hold on it."

Rule
DVD discs and Blu-ray Discs are video items.
Video items may be checked out for one week at a time.
Video items may be renewed up to two times.
Renewing a checked-out video item extends the due date by three days.
A patron may not renew an item that another patron has on hold.

Documenting Business Rules

- Apply structured templates for defining rules of different types.
- "A Unique Identifier": Link requirements back to a specific rule.
- "Rule Definition": Define the details of business rules.
- "Type of Rule": Identify each business rule of different type.
- "Static or Dynamic": Indicate how likely the rule is to change over time.
- "Sources": Identify the source of each rule: management policies, experts, documents.

Documenting Business Rules

- **ID**: ORDER-5
- **Rule Definition**:
  If the customer ordered a book by an author who has written multiple books, then offer the author's other books to the customer before completing the order.
- **Type of Rule**: Action Enablers
- **Static or Dynamic**: Static
- **Source**: Marketing policy xx

Documenting Business Rules

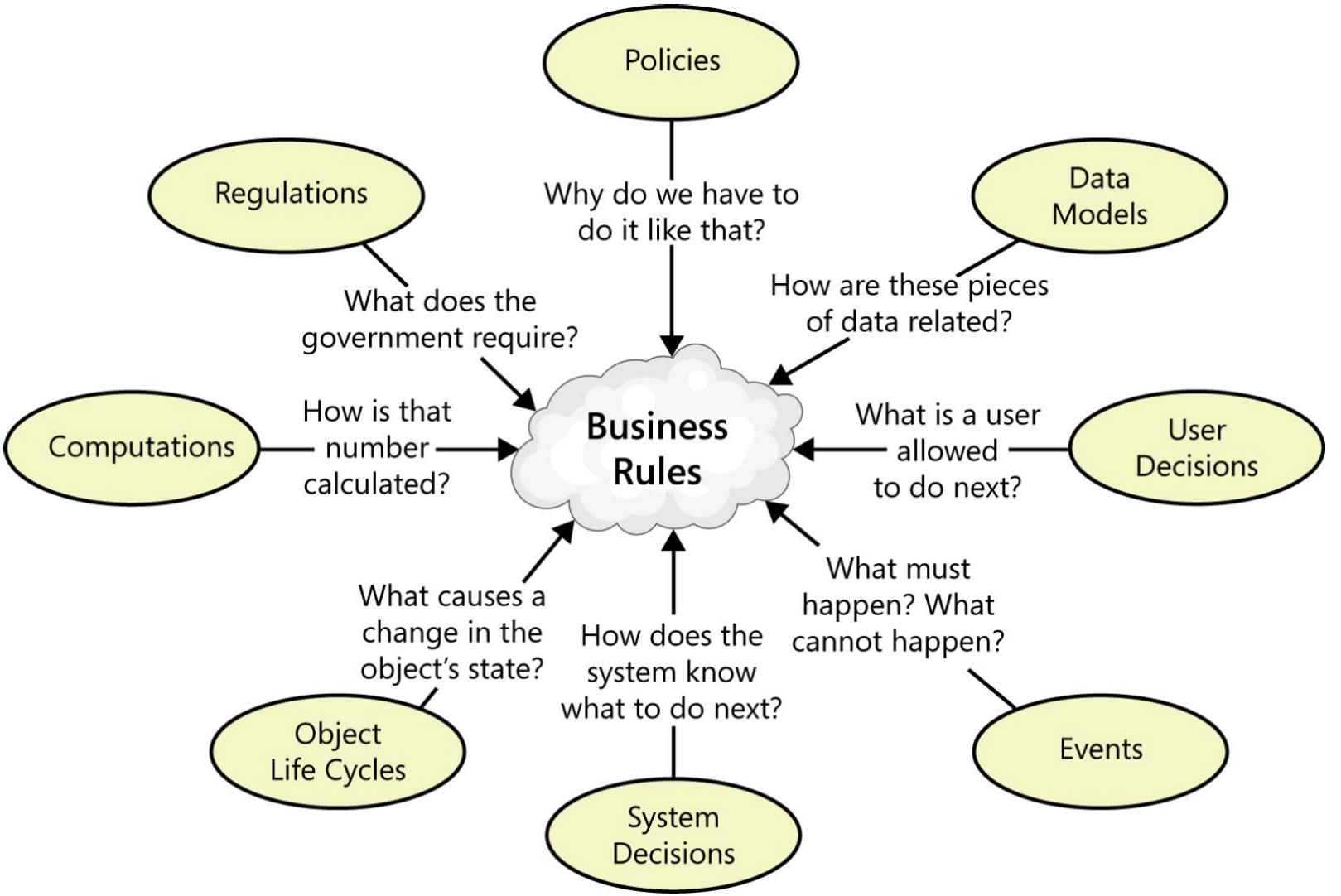
- **ID**: ACCESS-8
- **Rule Definition**:
  All website images must include alternative text to be used by electronic reading devices to meet accessibility requirements for visually impaired users.
- **Type of Rule**: Constraint
- **Static or Dynamic**: Static
- **Source**: ADA Standards for Accessible Design

ID	Rule definition	Type of rule	Static or dynamic	Source
ORDER-5	If the customer ordered a book by an author who has written multiple books, then offer the author's other books to the customer before completing the order.	Action enabler	Static	Marketing policy XX
ACCESS-8	All website images must include alternative text to be used by electronic reading devices to meet accessibility requirements for visually impaired users.	Constraint	Static	ADA Standards for Accessible Design

Discovering Business Rules

- "Common knowledge" from the organization collected from individuals
- Legacy systems that embed business rules in their requirements and code
- Business process modeling, to look for rules that can affect each process step
- Analysis of existing documentation
- Analysis of data, such as the various states that a data object can have and the conditions under which a user or system event can change the object's state.
- Compliance departments in companies

Discovering Business Rules



Business Rules and Requirements

- Determine which ones must be implemented in the software.
- The rules are external statements of policy that must be enforced in software, thereby driving system functionality.

The training records is current before a user can request a hazardous chemical.

- Look up the training records database if it is accessible;
- Query the training coordinator if it is inaccessible.

Business Rules and Requirements

The training records is current before a user can request a hazardous chemical.

- Look up the training records database if the user is trained;
- Query the training coordinator if the user is not found in the training records database.

FR-1: The system shall check the training records to find whether the user is trained before a user can request a hazardous chemical.
FR-2: The system shall query the training coordinator whether the user is trained before a user can request a hazardous chemical.
FR-3: The system shall reject the request if …

Example

Rule #1 (action enabler): "If the expiration date for a chemical container has been reached, then notify the person who currently possesses that container."
Rule #2 (fact): "A container of a chemical that can form explosive decomposition products expires one year after its manufacture date."

Example

Expired.Notify.Before *If the status of a chemical container that has an expiration date is not Disposed, the system shall notify the container's current owner one week before the date the container expires.*

Expired.Notify.Date *If the status of a chemical container that has an expiration date is not Disposed, the system shall notify the container's current owner on the date the container expires.*

Example

Expired.Notify.After *If the status of a chemical container that has an expiration date is not Disposed, the system shall notify the container's current owner one week after the date the container expires.*

Expired.Notify.Manager *If the status of a chemical container that has an expiration date is not Disposed, the system shall notify the manager of the container's current owner two weeks after the date the container expires.*

Example

Expired.Notify. *If the status of a chemical container that has an expiration date is not Disposed, the system shall notify the individuals shown in the following table at the times indicated.*

Requirement ID	Who to notify	When to notify
.Before	Container's current owner	One week before expiration date
.Date	Container's current owner	On expiration date
.After	Container's current owner	One week after expiration date
.Manager	Manager of container's current owner	Two weeks after expiration date

Review

- Business Rules Categories
  - Facts
  - Constraints
  - Action Enablers
  - Inferences
  - Computations
- Atomic Business Rules
- Documenting Business Rules
- Discovering Business Rules
- Software Requirements from Business Rules

Outline

- Understanding User Requirements
- Understanding Business Rules
- Specifying Data Requirements

Specifying Data Requirements

- Software manipulates data in ways that provide value to customers
  - Create, modify, display, delete, process, and use data
- To determine what data is required for building the model
  - The input and output flows on context diagram
  - Data flows represent major data elements at a high level of abstraction
- The process used to identify, prioritize, precisely formulate, and validate the data needed to achieve business objectives

Specifying Data Requirements

- A data entity is an object in a data model
  - Customer, address, book
- Data Attributes: a single descriptor for a data object
  - Customer: age, gender, weight, occupation…
  - Address: apartment No. Street, no. road name, post code.,…
  - Book: title, author, series id, price, year….
- Data Process: activities that transform data
  - Enter address, change address, delete address, …

- **Model data processes that take place in a system**
- **Depict the relationship between data entities**
- **Define the detailed data entities**

Specifying Data Requirements

- Data Flow Diagram (DFD)
  - A big-picture view of how data moves through a system
  - Show data processes that take place in a system
- Entity-relationship Diagram (ERD)
  - The high-level view of data entities in a system
  - Depict the relationship between data entities
- Data Dictionary
  - A collection of detailed information about data entities
  - Define the detailed data entities

Data Flow Diagram

- Data flow modeling takes a functional decomposition approach to systems analysis, breaking complex problems into progressive levels of detail.
- To identify:
  - The transformational processes of a system
  - The collections (stores) of data that the system manipulates
  - The flows of data between processes, stores, and the outside world.

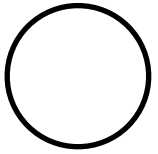
DFD Concepts

- Data Process (data transformation)
  - activities that transform data
- Data Flow
  - flow of data from one entity to another
  - represent a data structure or data element
- Data Store
  - place where data is held for later use
  - passive: no transformations happen here

DFD Concepts

- External Entity
  - an entity outside the target system
  - can only interact with processes
- Data Structure (Data Group)
  - cluster of data represented as a single data flow
  - composed of multiple data groups or elements
- Data Element
  - basic unit of data

Data Flow Diagrams



Process

Data Flow

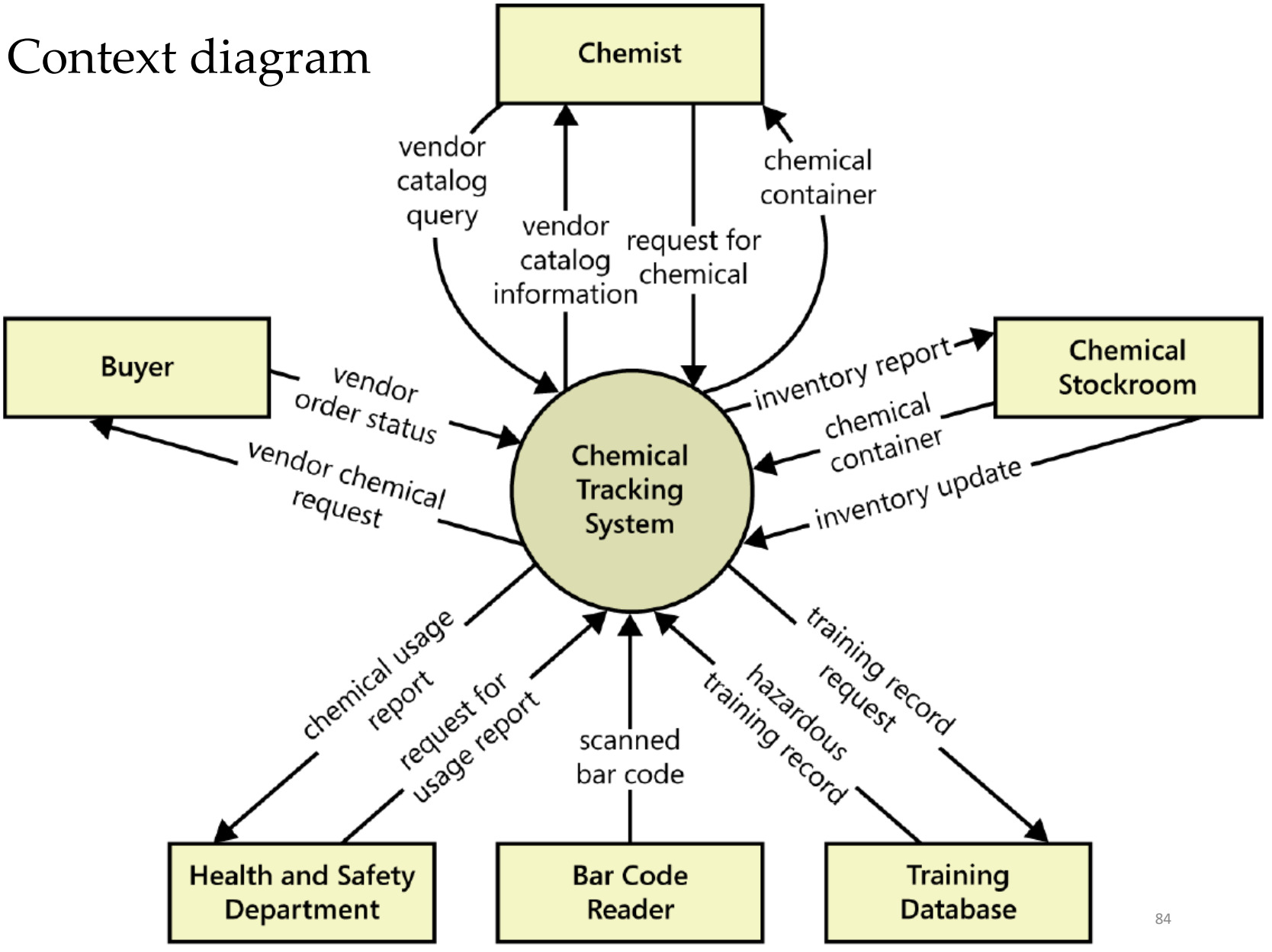
Data Store

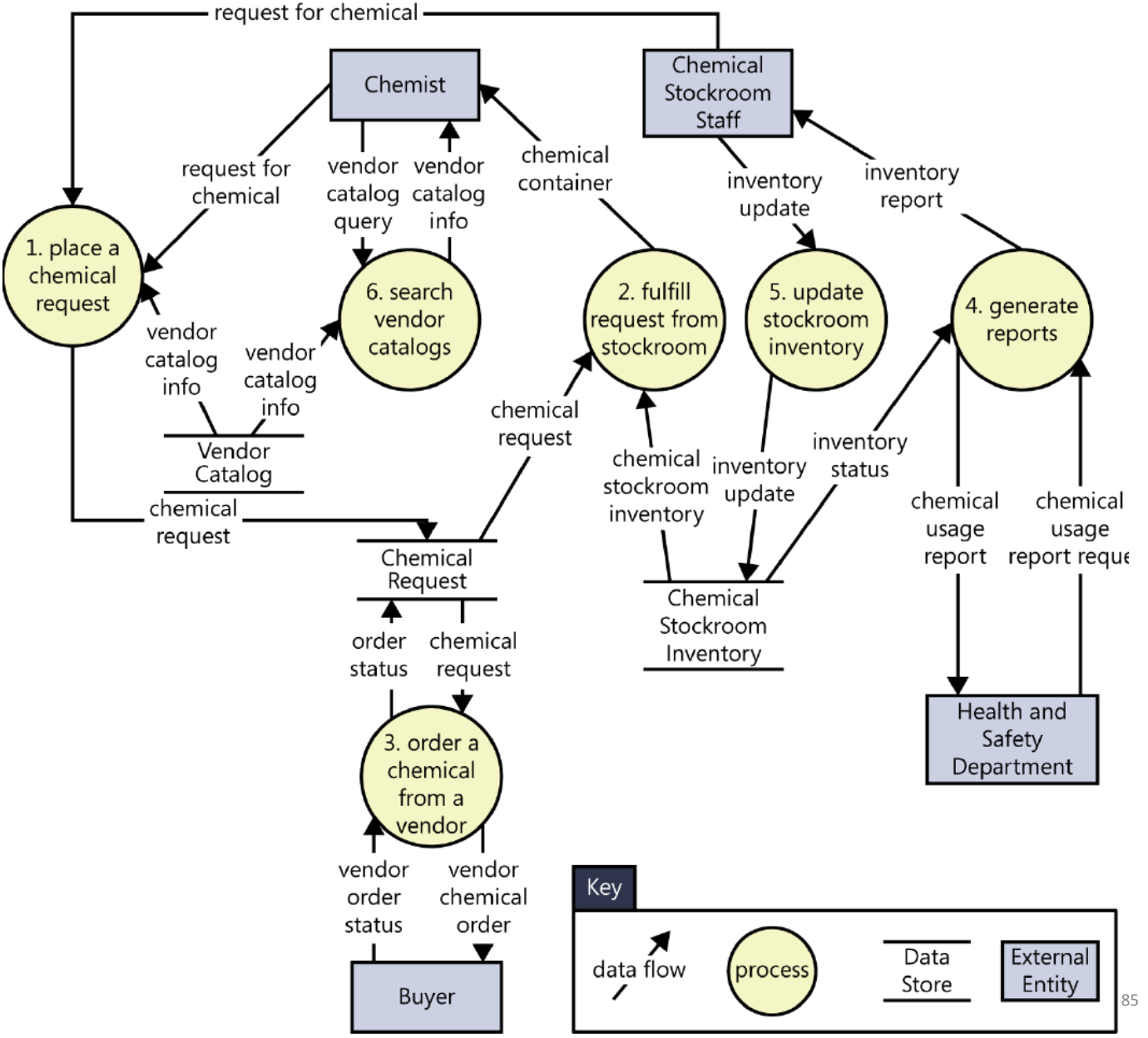
External Entity (terminator)

System Boundary

Every process, flow and data store must be labeled

Context diagram




request for chemical
Chemist
Chemical Stockroom Staff
request for chemical
vendor catalog query
vendor catalog info
chemical container
inventory update
inventory report
1. place a chemical request
6. search vendor catalogs
2. fulfill request from stockroom
5. update stockroom inventory
4. generate reports
vendor catalog info
vendor catalog info
Vendor Catalog
chemical request
chemical request
chemical stockroom inventory
inventory update
inventory status
chemical usage report
chemical usage report reque
Chemical Request
Chemical Stockroom Inventory
order status
chemical request
Health and Safety Department
3. order a chemical from a vendor
vendor order status
vendor chemical order
Buyer
Key
data flow
process
Data Store
External Entity

Data Flow Diagram

- Used when interviewing customers
- A technique to identify missing data requirements
- Give context to the functional requirements regarding how the user performs specific tasks
- The context diagram represents the highest level of abstraction of the DFD
- Each process that appears as a separate bubble on the high-level DFD diagram can be further expanded into a low-level DFD.

DFD Conventions

- Processes communicate through data stores, not by direct flows from one process to another.
- Data cannot flow directly from one store to another or directly between external entities and data stores;
- Data flow must pass through a process bubble.
- Don't **attempt to imply the processing sequence** using the DFD.
- Name each process as a concise action: **verb plus object**. Use names that are meaningful to the customers and pertinent to the business

DFD Conventions

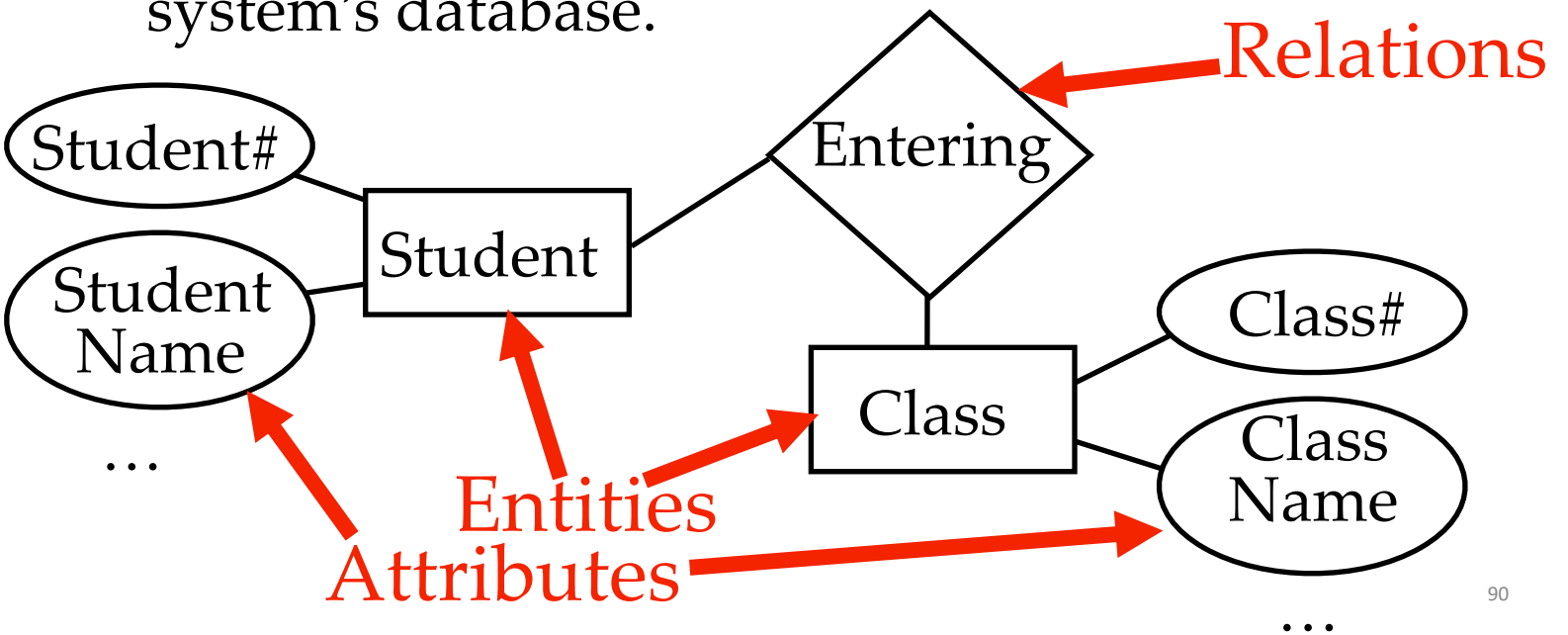
- Number the processes uniquely and hierarchically. Number each process with an integer on the level 0 diagram
- Don't show more than 8 to 10 processes on a single diagram or it will be difficult to draw, change, and understand
- If you have more processes, introduce another layer of abstraction by grouping related processes into a higher-level process
- Bubbles with flows that are only coming in or only going out are suspect

Review

- Data Requirements
- Data Entity, Data Attribute, Data Process
- Data Flow Diagram
  - Process, Data Flow, Data Store, External Entity Data Structure, Data Element
  - DFD Concepts
  - DFD Conventions

Entity-Relationship Diagram

- A requirements analysis tool
  - Understand and communicate data entities of the business or the system
  - Model the relationships of data entities in a system
  - Define the logical or physical structure of the system's database.



Entity-Relationship Diagram

- **Rectangles**: Entities that represent physical items or aggregations of data.
- Entities are named as singular nouns, e.g., Student, and Class.
- **Oval**: Attributes that an entity has
- An entity has multiple attribute values.

  The attributes for each student include a unique ID, the name, the gender, the subject, and the department.
- Data dictionary contains the precise definitions of the attributes.

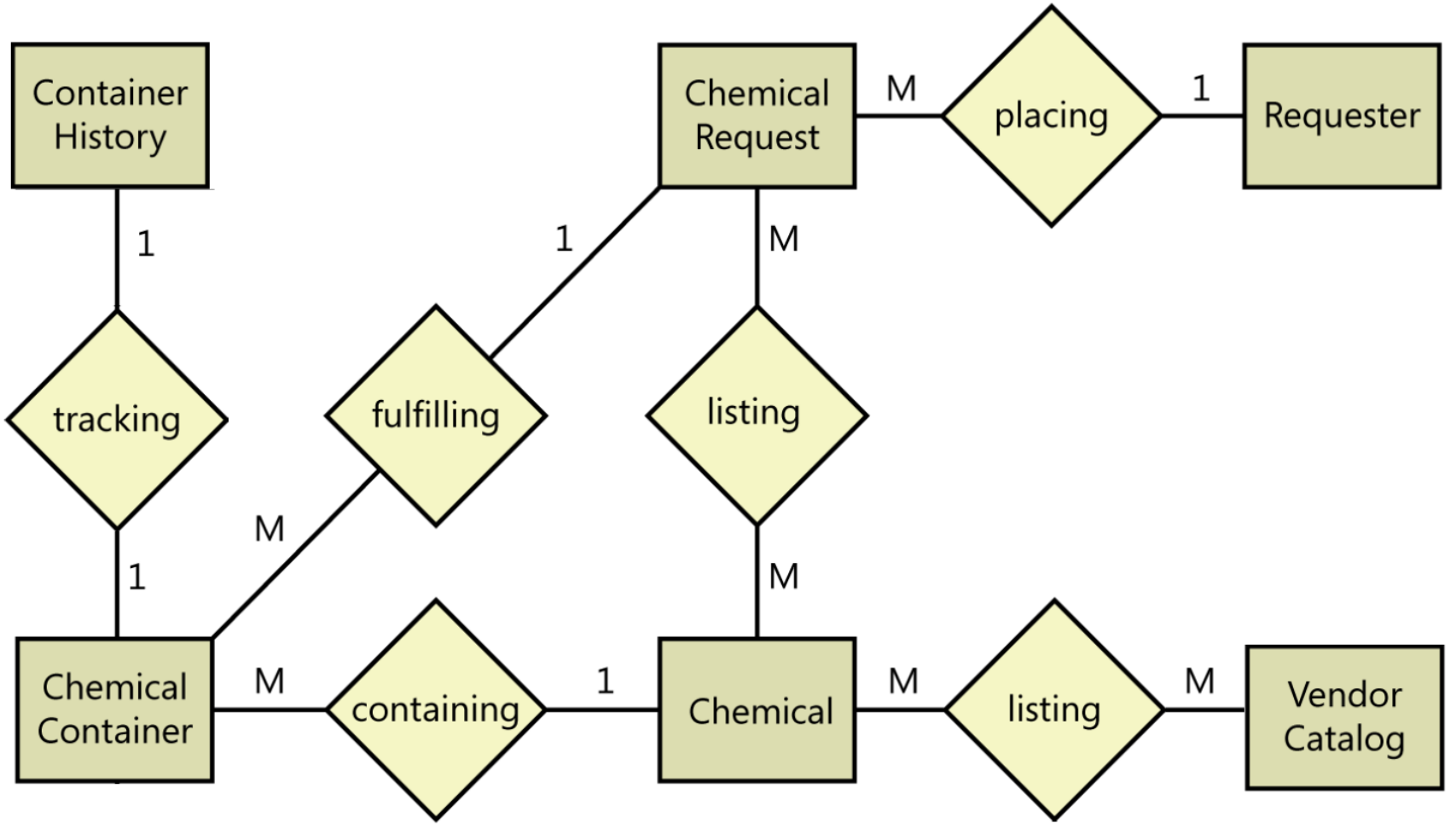
Entity-Relationship Diagram

- **Diamonds**: Relationships that identify the logical linkages between pairs of entities.
- Describe the nature of the connection

  The relationship between the student and the class is an entering relationship

- Name the relationship "doing".

Entity-Relationship Diagram



Entity-Relationship Diagram

- **Cardinality/multiplicity**: A number or letter on the lines that connect entities and relationships.
  - ***One-to-many* relationship**: A requester can *place* multiple Chemical Requests.
  - ***One-to-one* relationship**: Every Chemical Container is tracked by a single Container History
  - ***Many-to-many* relationship**: Every Vendor Catalog lists many Chemicals, and some Chemicals are listed in multiple Vendor Catalogs.

Entity-Relationship Diagram

- Use different symbols on the lines connecting entities and relationships to indicate cardinality.
- The relationship between entities is labeled on the line that connects the entities.
- The vertical line indicates a cardinality of 1,
- The crow's foot symbol indicates a cardinality of many.

Entity-Relationship Diagram

- Widely used in information modelling
- Simple, easy to use
  – notation, not method
- Used in many contexts
  – data to be represented in the system
- Express relationships between data flows in different data stores.
- Check whether the relationships shown are all correct.
- Identify any missing entities or any possible relationships between entities.

UML Class Diagram

- Show the data attributes for individual classes (entities in the ERD), the logical links between classes, and the cardinalities of those links
- A class is shown in a rectangle
  - List the attributes associated with each class in the middle section of the rectangle
- 1..* notation means one or more

Data Dictionary

- A collection of detailed information about the data entities used in a system.
- Collect the information about composition, data types, allowed values, and the like into a shared resource.
- Represent data elements and data structures.
- Feed into design in the form of database schemas, tables, and attributes

Data Dictionary Advantages

- Identify the data validation criteria.
- Avoid the mistakes that can result when project participants have different understandings of the data.
- Help developers write programs correctly
- Minimize integration problems

Data Dictionary Template

- Data Element

  Primitive data element and data structure in the system
- Description

  The description of data element
- Composition or Data Type

  Data type: integer, numeric, alphabetic characters
- Length

  The size of data element
- Values

  The description of the values of data element

Example Data Dictionary

Data Element	Description	Composition or Data Type	Length	Values
Chemical Request	request for a new chemical from either the Chemical Stockroom or a vendor	Request ID + Requester + Request Date + Charge Number + 1:10{Requested Chemical}		
Delivery Location	the place to which requested chemicals are to be delivered	Building + Lab Number + Lab Partition		
Number of Containers	number of containers of a given chemical and size being requested	Positive integer	3	
Quantity	amount of chemical in the requested container	numeric	6	
Quantity Units	units associated with the quantity of chemical requested	alphabetic characters	10	grams, kilograms, milligrams, each
Request ID	unique identifier for a request	integer	8	system-generated sequential integer, beginning with 1

Data Dictionary –Data Elements

- **Primitive (element):** no further decomposition is possible or necessary.
e.g. Number of Containers, Quantity, Quantity Units, and Request ID

- **Structure:** A data structure is composed of multiple data elements with plus (+) signs.
e.g., Chemical Request, Delivery Location

Data Dictionary-Example

- Data Element

  Number of Containers
- Description

  Number of containers of a given chemical and size being requested
- Composition or Data Type

  Positive Integer
- Length

  3
- Values

  N/A

Data Dictionary-Example

- Data Element

  Chemical Request
- Description

  Request for a new chemical from either the Chemical Stockroom or a vendor
- Composition or Data Type

  Request ID

  + Requester

  + Request Date

  + Charge Number

  + 1:10{Requested Chemical}

Data Dictionary

Repeating group: Multiple instances of a data element can appear in a structure

- *1:10{Requested Chemical}*: A chemical request must contain at least one chemical but may not contain more than 10 chemicals.
- *3:n{Requested Chemical}:* If the maximum number of instances in a repeating field is unlimited, use "*n*" to indicate this.
- Repetition $^{\min}\{\}^{\max}$: $^{3}\{\text{numeric}\}^{10}$
  - no min means constant size: $\{\text{numeric}\}^{16}$

Data Dictionary

Optimal: An element in data structure is optional.

- A value doesn't have to be supplied by the user or the system.

Hyperlinks: Navigation links in a data dictionary

- Span many pages, or even multiple documents if a project's data dictionary incorporates some definitions from an enterprise-wide data dictionary.

Data Dictionary-Example

- Data Element

  Requested Chemical
- Description

  Description of the chemical being requested
- Composition or Data Type

  Chemical ID

  + Number of Containers

  + Grade

  + Quantity

  + Quantity Units

  + (Vendor)

Data Dictionary

- Data definitions are reusable across applications, particularly within a product.
- Using consistent data definitions across the enterprise reduces integration and interface errors.
- A separate data dictionary makes it easy to find the information you need.
- Avoid redundancies and inconsistencies.

Review

- Entity-Relationship Diagram: Functionalities
  - Attributes (Oval), Entities (Rectangles) and Relations (Diamonds)
  - Name the relationship "doing"
  - Cardinality/multiplicity
  - Other Symbols: vertical line, crow's foot symbol
- Data Dictionary: A collection of detailed information about the data entities used in system
  - Five Entries in Template
  - Primitive and Structure
  - Repeating Group, Optimal, Hyperlinks