Software Modeling & Analysis

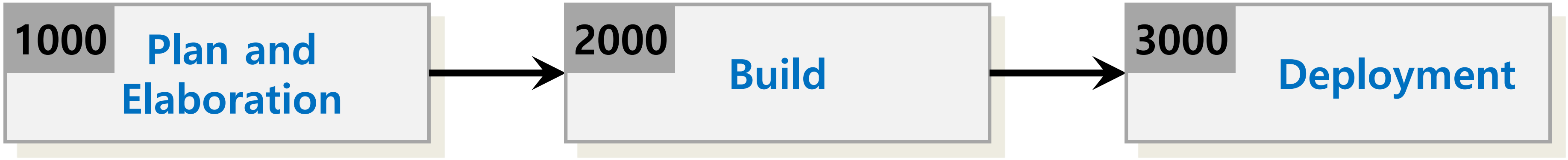
OOPT (Object Oriented Process with Trace)

Lecturer: JUNBEOM YOO
jbyoo@konkuk.ac.kr

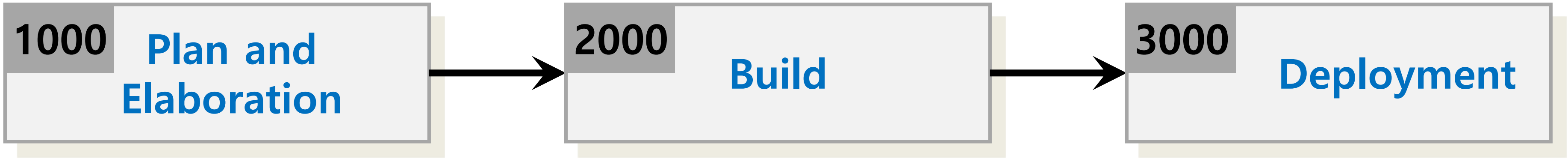
What is OOPT?

- OOPT (Object Oriented Process with Trace)
  - A software process based on RUP
  - Revision of OSP (by Tailored to SE classes in universities)

- Characteristics of OOPT
- 3 Stages
  1. Iterative : Multiple development cycles
  2. Incremental : System grows incrementally as each cycle is completed
  3. Architecture : Stage > Cycle > Phase > Activity



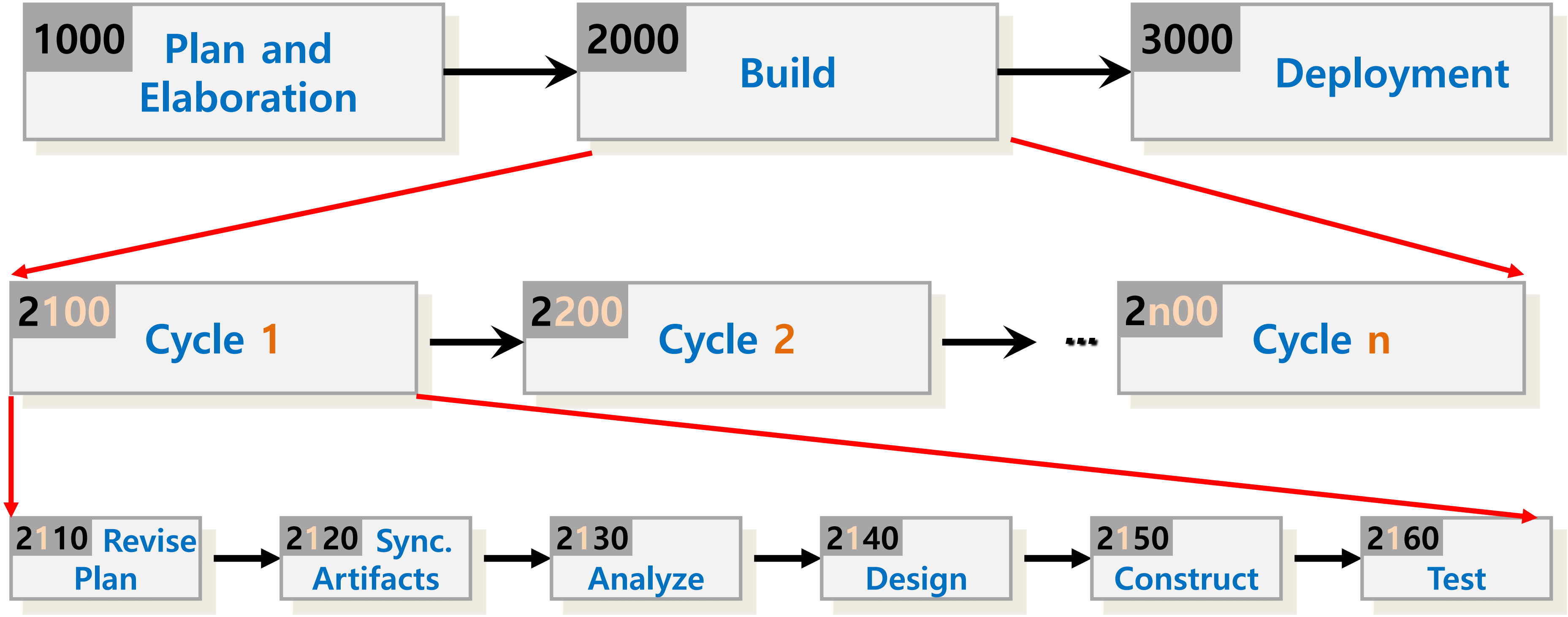
1. 3 Stages



- Stage 1000 : Plan and Elaboration
  - Planning, defining requirements, building prototyping, etc
  - Corresponding to Inception/Elaboration phases in the RUP
- Stage 2000 : Build
  - Construction of the system
  - Corresponding to Construct phase in the RUP
- Stage 3000 : Deployment
  - Implementation of the system into use
  - Corresponding to Transition phase in the RUP

2. Iterative Development

- Multiple iterations in the Build stage
- Each iteration took about 2 to 8 weeks



1000 Plan and Elaboration

2000 Build

3000 Deployment

2100 Cycle 1

2200 Cycle 2

…

2n00 Cycle n

2110 Revise Plan

2120 Sync. Artifacts

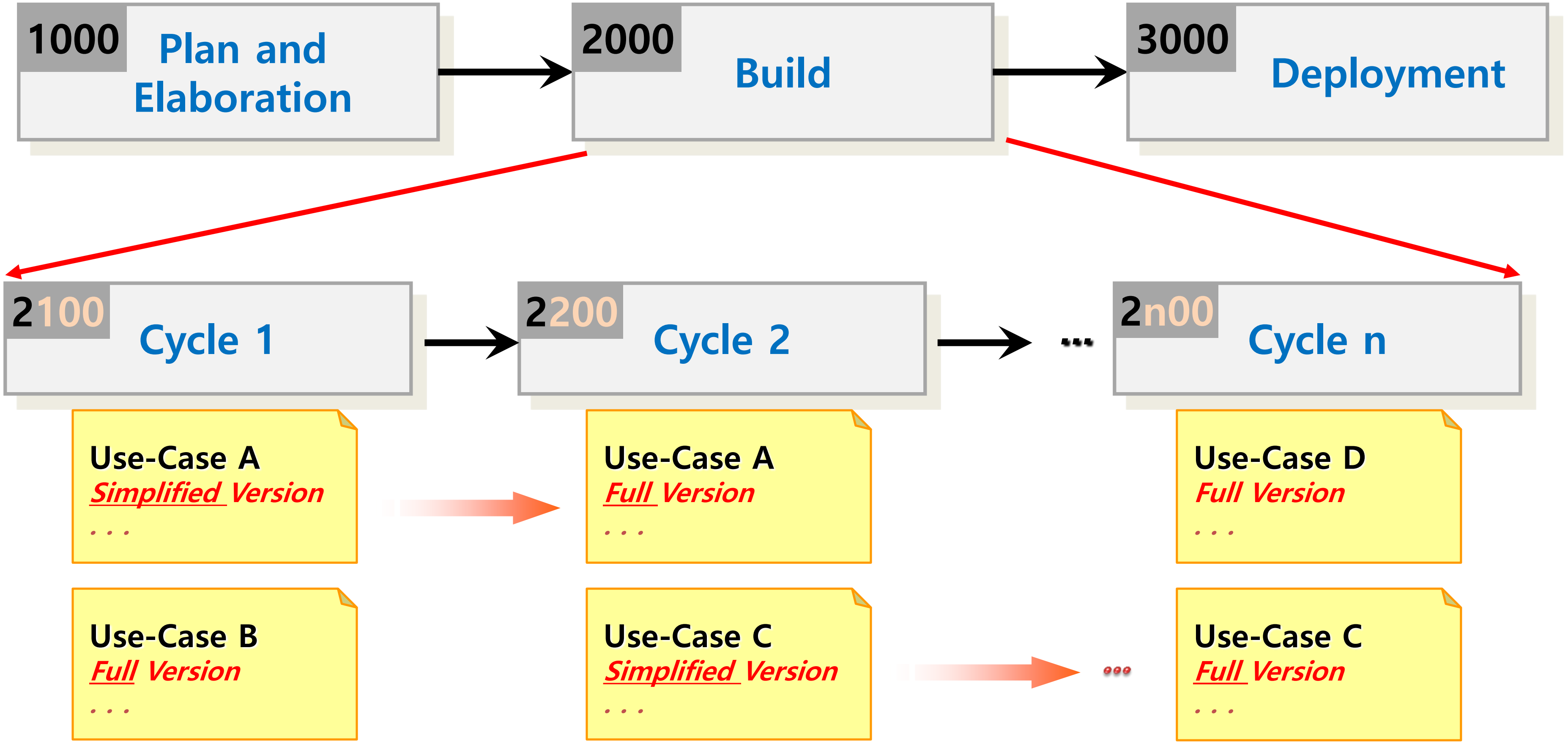
2130 Analyze

2140 Design

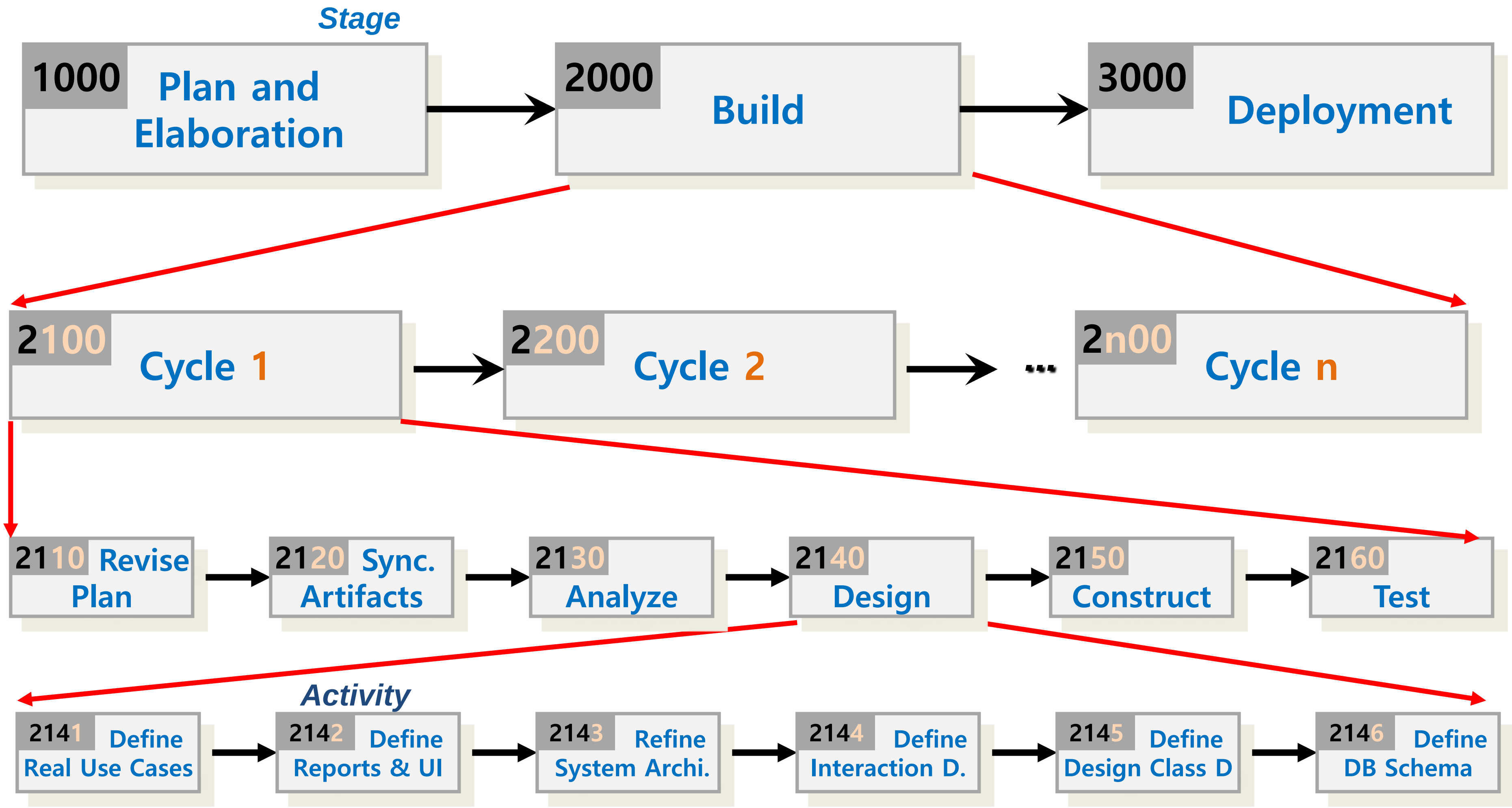
2150 Construct

2160 Test

3. Incremental Development

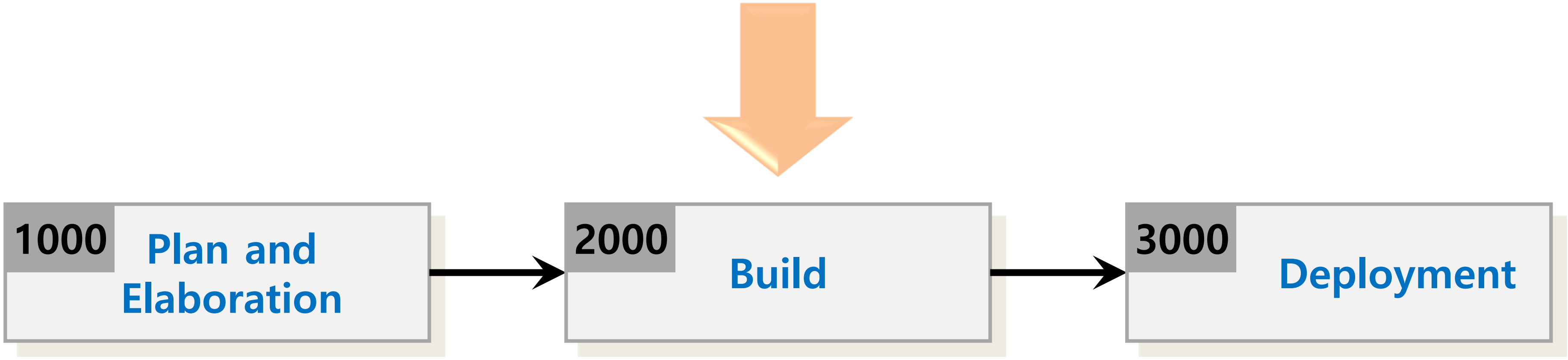


4. Architecture of OSP



Dependable Software Laboratory

Stage 2000. Build

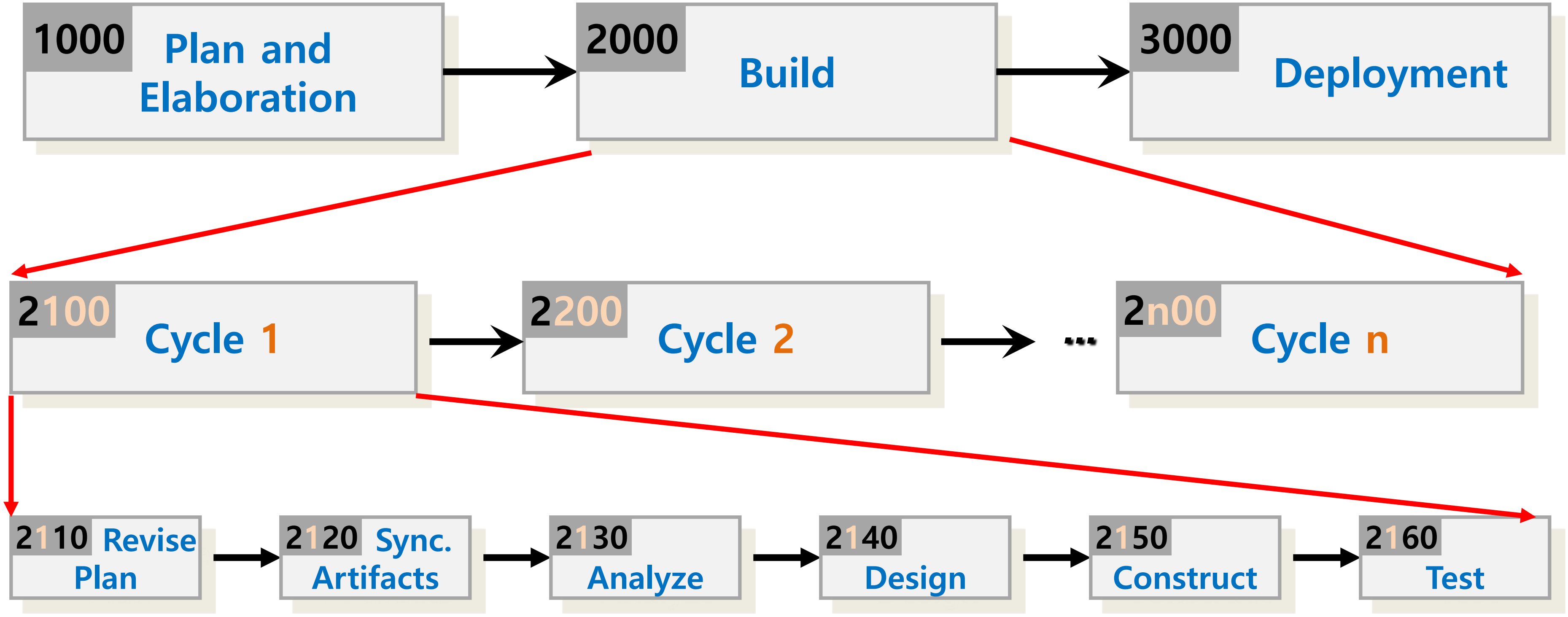


1000 Plan and Elaboration

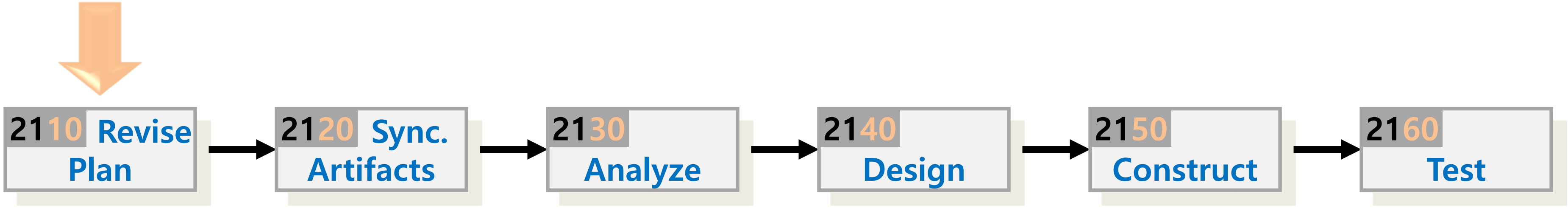
2000 Build

3000 Deployment

6 Phases of ‘Build’ Stage



Phase 2010. Revise Plan



2110 Revise Plan

2120 Sync. Artifacts

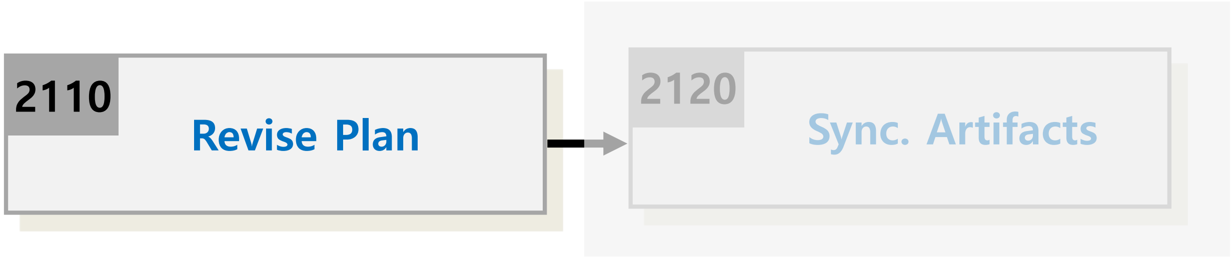
2130 Analyze

2140 Design

2150 Construct

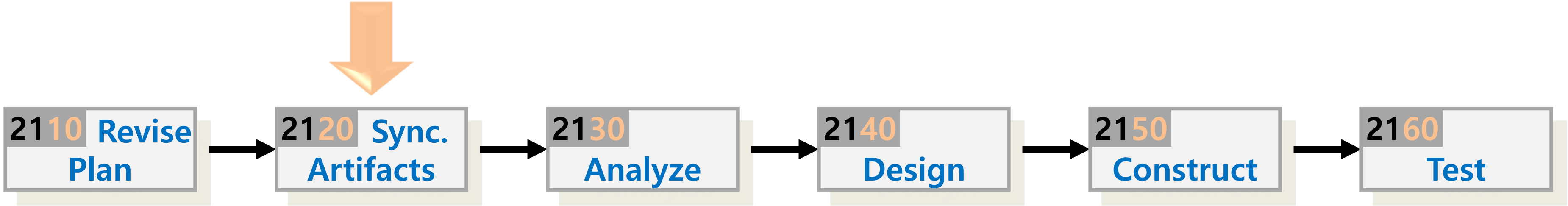
2160 Test

Phase 2010. Revise Plan



- Description
  - Correct and enhance the project plan and requirement definition based on the intermediate deliverables
  - Input : intermediate deliverables
  - Output : A refined project plan, a refined requirement specification
- Steps

Phase 2020.
Synchronize Artifacts



2110 Revise Plan

2120 Sync. Artifacts

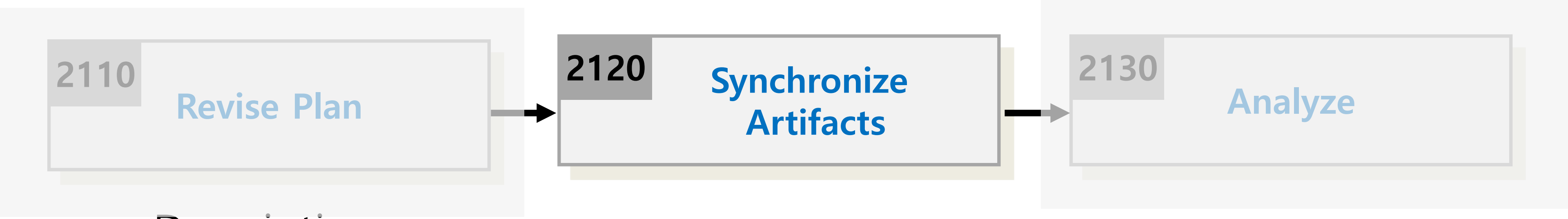
2130 Analyze

2140 Design

2150 Construct

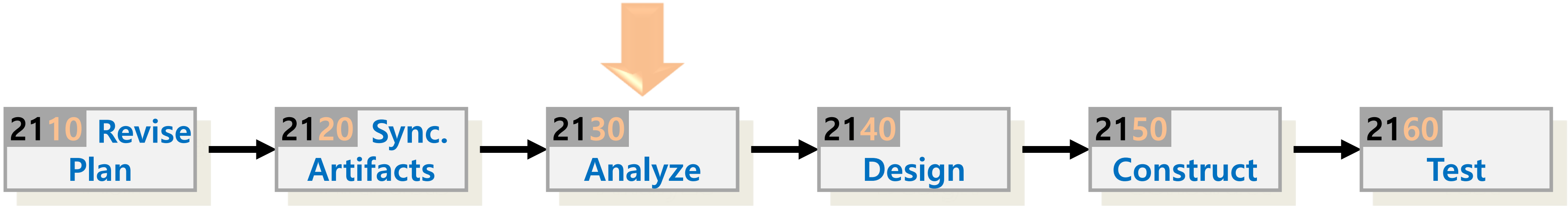
2160 Test

Phase 2020. Synchronize Artifacts



- Description
  - Configure and manage various types of artifacts (Project Repository)
  - Control versions and variations
  - Input :
  - Output :
- Steps

Phase 2030. Analyze



2110 Revise Plan

2120 Sync. Artifacts

2130 Analyze

2140 Design

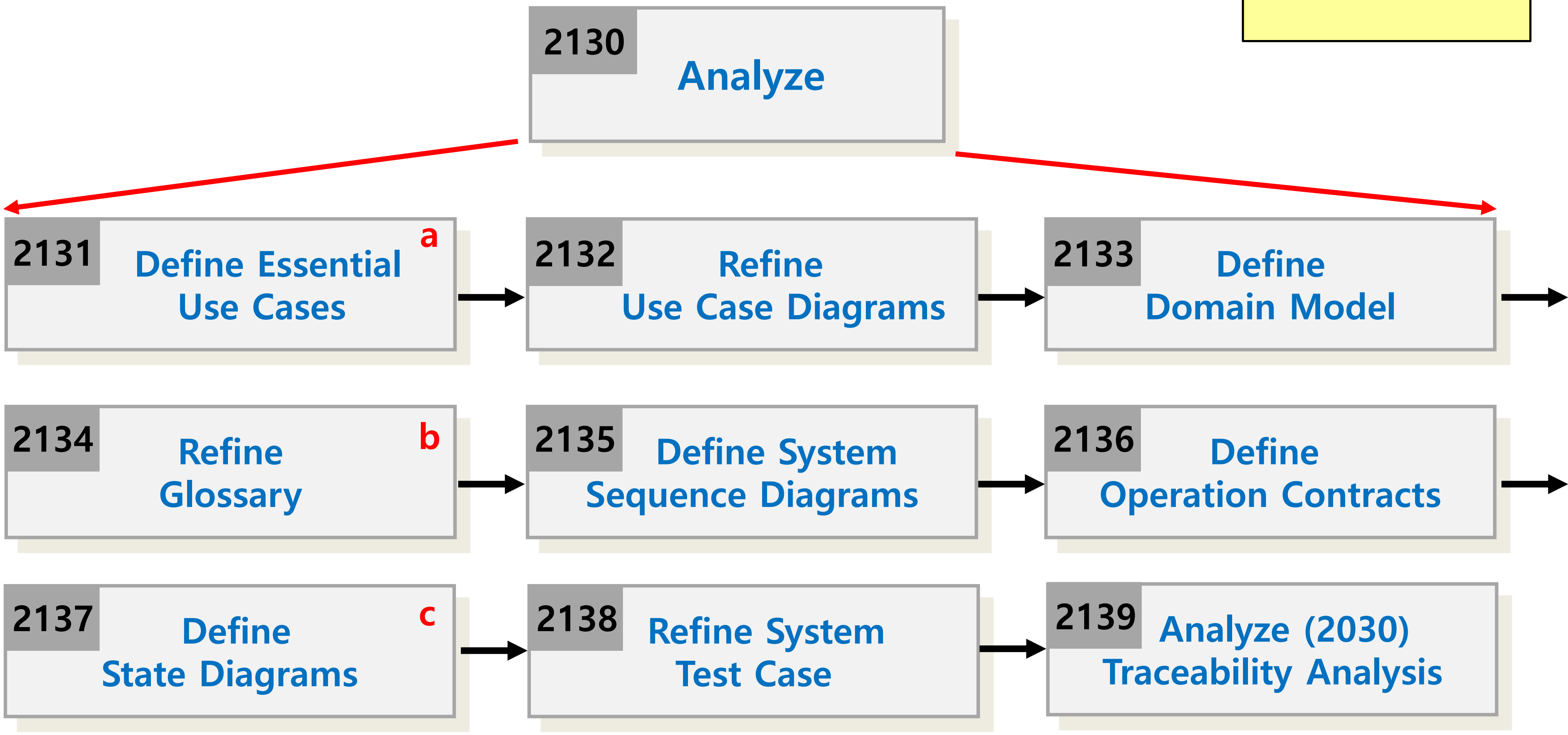
2150 Construct

2160 Test

Phase 2030. Analyze

- Phase 2030 Activities

a. if not yet done
b. ongoing
c. optional



2130 Analyze

2131 Define Essential Use Cases a

2132 Refine Use Case Diagrams

2133 Define Domain Model

2134 Refine Glossary b

2135 Define System Sequence Diagrams

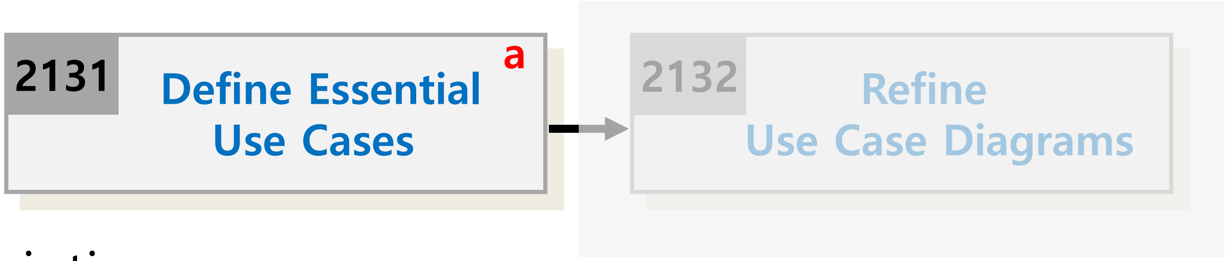
2136 Define Operation Contracts

2137 Define State Diagrams c

2138 Refine System Test Case

2139 Analyze (2030) Traceability Analysis

Activity 2031. Define Essential Use Cases



- Description
  - Add event flows to business use case(high-level) descriptions
  - Input : business use case descriptions (activity 1006)
  - Output : An essential use case descriptions
  - Standard applied : expanded use case format

Activity 2031. Define Essential Use Cases

- Step
  1. Select each use case from business use cases
  2. Identify system functions related to the selected use case from requirements specification
  3. Identify related use cases to the selected use case from business use cases
  4. Identify courses of events for each use case from the requirements specification
     - Typical courses of events (main event flow)
     - Alternative courses of events
     - Exceptional courses of events
  5. Write essential use cases based on typical and alternative courses of events flows by applying expanded use case format.

Activity 2031. Define Essential Use Cases

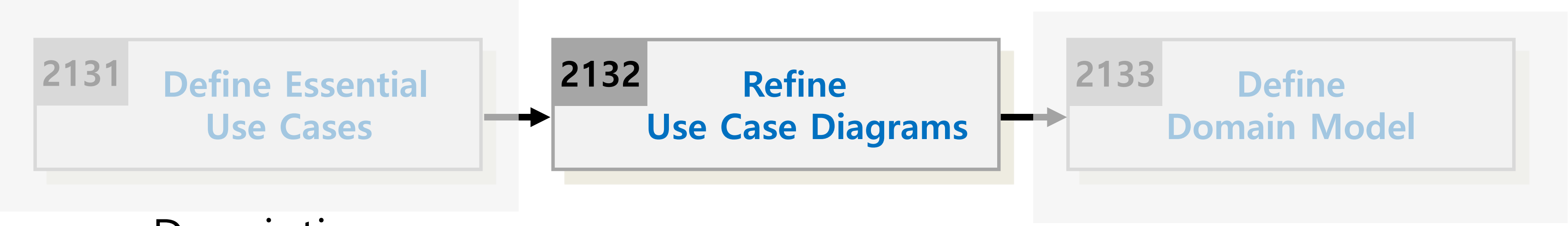
- Expanded Use Case Format
  - Use case: Use Case Name
  - Actors: Actor Name
  - Purpose: The purpose of Use Case
  - Overview: The overview of Use Case
  - Type: Primary, Secondary, or Optional
  - Cross References: System functions in Req. Spec
  - Pre-Requisites: An essential pre-condition
  - Typical Courses of Events: Abstract scenario about the flow of events
  - Alternative Courses of Events:
  - Exceptional Courses of Events: define exceptional cases

Activity 2031. Define Essential Use Cases

- Example: “Buy Items”

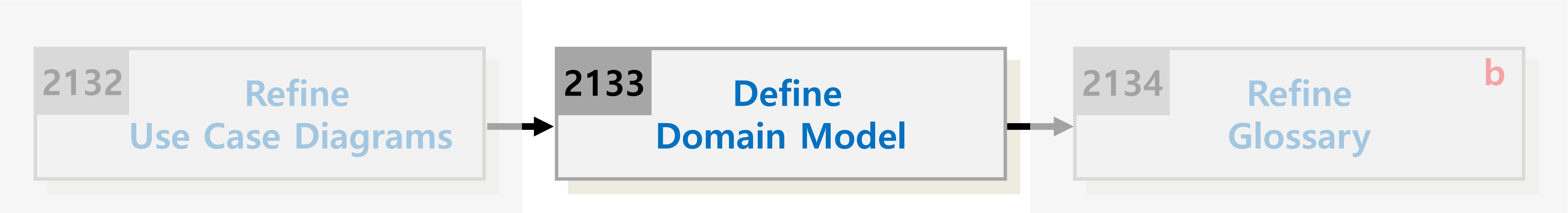
Use Case	Buy Items
Actor	Customer, Cashier
Purpose	Capture a sale and its payment
Overview	A Customer arrives at a checkout with items to purchase. The Cashier records the items and collects a payment, which may be authorized. On completion, the Customer leaves with the items.
Type	Primary and Essential
Cross Reference	Functions: R1.1, R1.2, R1.3, R1.7, R1.9, R2.1, R2.2, R2.3, R2.4 Use Cases: Log In use case
Pre-Requisites	N/A
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) This use case begins when a customer arrives at the POST to checkout with items to purchase. 2. (A) The Cashier records each item.(E1) 3. (S) Determines the item price and adds the item information to the running sales transaction. 4. (A) On completion of item entry, the cashier indicates to the POST that item entry is complete. 5. (S) Calculates and presents the sale total. 6. (A) The Cashier tells the customer the total.
Alternative Courses of Events	…
Exceptional Courses of Events	E1: If invalid item identifier entered, indicate error.

Activity 2032. Refine Use Case Diagrams



- Description
  - Validate and modify the 'Business Use-Case Diagram'
  - Input : business use case model, essential use case descriptions
  - Output : A refined use case diagram
  - Standard applied : UML's use case diagram
- Step
  1. Review business use case diagrams according to essential use case descriptions
  2. Refine use case diagrams by adding or refining use cases and relationships

Activity 2033. Define Domain Model



- Description
  - Define domain concept model by reviewing input artifacts
  - Input : essential use case descriptions, business concept model
  - Output : A conceptual class diagram
  - Standard applied : UML's use case diagram

- What is domain model?
  - A representation of conceptual classes identified from a real world
  - Illustrates meaningful conceptual classes in a problem domain.
  - Conceptual models
  - Widely used as a source of inspiration for designing software objects.

Activity 2033. Define Domain Model

- Step
  1. List concepts(domain class) from use cases or business concept model
     - Guideline 1
       - Identify concepts by making a list of candidate concepts from the ‘Concept Category List’
     - Guideline 2
       - Identity the noun and noun phrases in expanded use cases description and consider them as candidate concepts or attributes

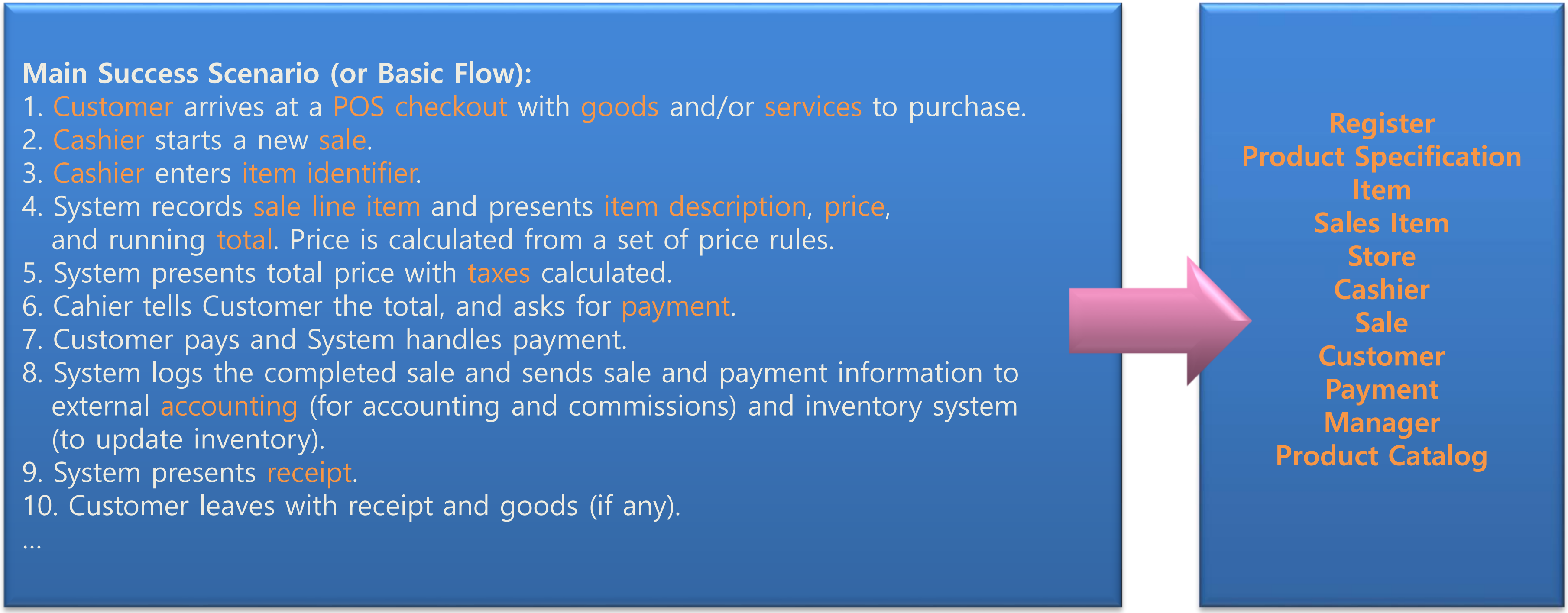
Activity 2033. Define Domain Model

- By using guideline 1
  - ‘Concept Category List’ may contain many common categories that are usually worth to consider

Concept Category	Examples
Physical or tangible objects	POST
Specifications, designs, or descriptions of things	Product Specification
Places	Store
Transactions	Sale Payment
Transaction line items	Sales Line Item
Roles of people	Cashier
Containers of other things	Store
Things in a container	Item
Other computer or electro-mechanical systems external to our system	Credict Card Authorization System
…	…

Activity 2033. Define Domain Model

- By using guideline 2
  - The fully dressed use cases are an excellent description
  - Scenario of the use case or use case descriptions can be used.



Main Success Scenario (or Basic Flow):
1. Customer arrives at a POS checkout with goods and/or services to purchase.
2. Cashier starts a new sale.
3. Cashier enters item identifier.
4. System records sale line item and presents item description, price, and running total. Price is calculated from a set of price rules.
5. System presents total price with taxes calculated.
6. Cahier tells Customer the total, and asks for payment.
7. Customer pays and System handles payment.
8. System logs the completed sale and sends sale and payment information to external accounting (for accounting and commissions) and inventory system (to update inventory).
9. System presents receipt.
10. Customer leaves with receipt and goods (if any).

…

Register
Product Specification
Item
Sales Item
Store
Cashier
Sale
Customer
Payment
Manager
Product Catalog

Activity 2033. Define Domain Model

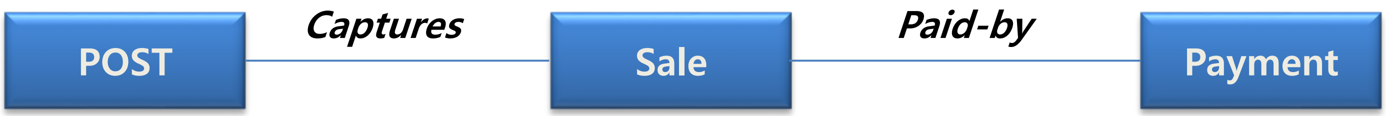
2. Assign class names into concepts
   - Use the existing names in the domain
   - Do not add things that are not there
3. Identify associations according to association categories

Association Category	Identified Associations
A is a physical part of B	Drawer – POST
A is a logical part of B	SalesLineItem – Sale
A is physically contained in/on B	POST – Store Item – Shelf
A is logically contained in B	ItemDescription – Catalog
A is a description for B	ItemDescription – Item
A is a line item of a transaction or report B	SalesLineItem – Sale
A is known/logged/recorded/reported/captured in B	Sale – POST
A is a member of B	Cashier –Store
A is an organizational submit of B	Department – Store
…	…

Activity 2033. Define Domain Model

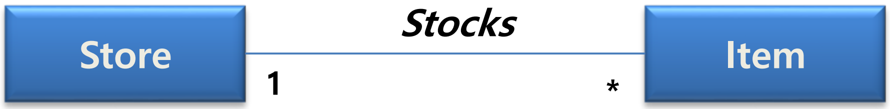
4. Assign priorities into identified associations
   - High priority association categories are
     - A is a physical or logical part of B.
     - A is physically or logically contained in/on B.
     - A is recorded in B.
   - Should avoid showing redundant or derivable associations

5. Assign names into associations
   - *"Type Name" – "Verb Phrase" – "Type Name"*
   - Association names should start with a capital letter.

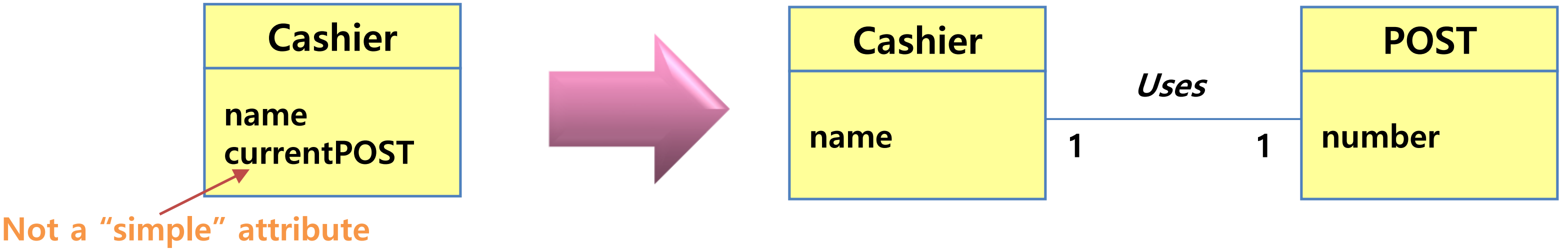


Activity 2033. Define Domain Model

6. Add multiplicity into the ends of an association

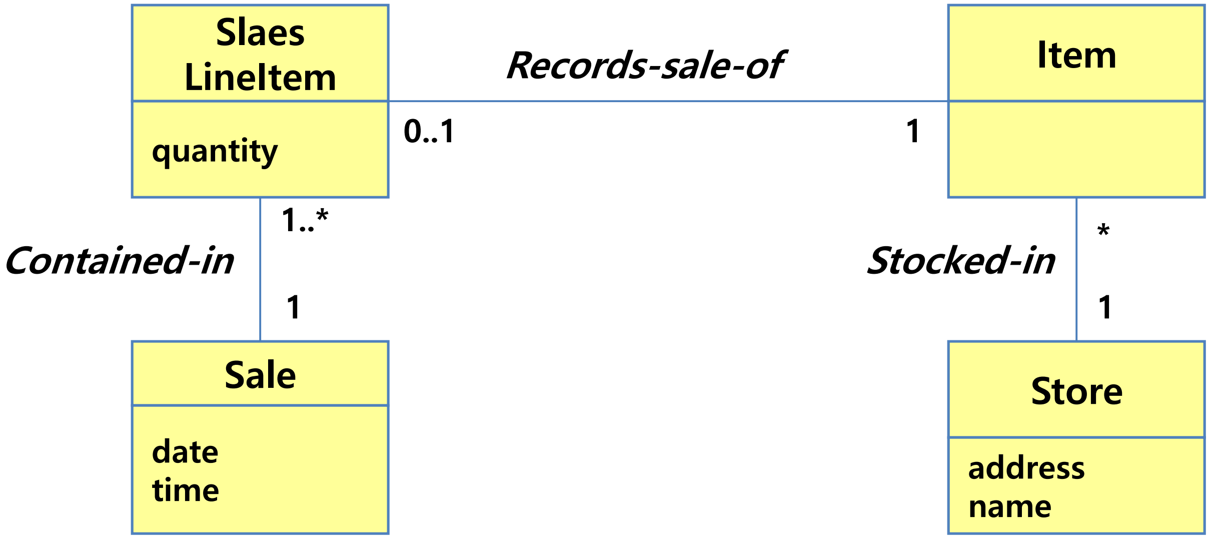


7. Identify attributes by reading
   - requirement specifications, current use cases under consideration, simplification, clarification, and assumption documents
   - Attributes should be simple attributes or pure data values
     - Boolean, Date, Number, String, Time
     - Address, Color, Geometrics(Point, Rectangle,…), Phone Number, Social Security Number, Universal Product Code(UPC), ZIP or postal codes, Enumerated types.

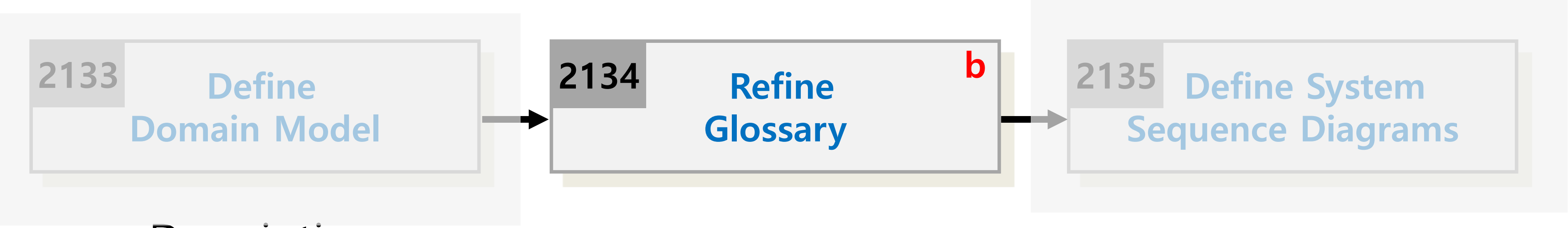


Activity 2033. Define Domain Model

8. Draw them in a conceptual class diagram



Activity 2034. Refine Glossary



2133 Define Domain Model

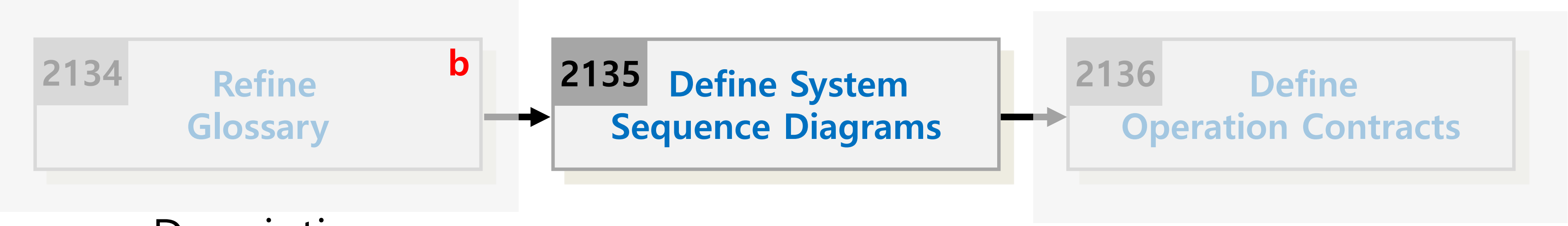
2134 Refine Glossary b

2135 Define System Sequence Diagrams

- Description
  - Lists and refines all the terms in order to improve communication and reduce the risk of misunderstanding
  - Input : term dictionary, essential use case descriptions, conceptual class diagram
  - Output : A refined term dictionary
- Step
  1. Refine terms defined in the Plan and Elaborate Phase(use cases, attributes, concept, etc.) during development cycle.
  2. Record terms as following format:

Term	Category	Comments
Payment	Concept (Class)	a cash payment
…	…	…

Activity 2035.
Define System Sequence Diagrams



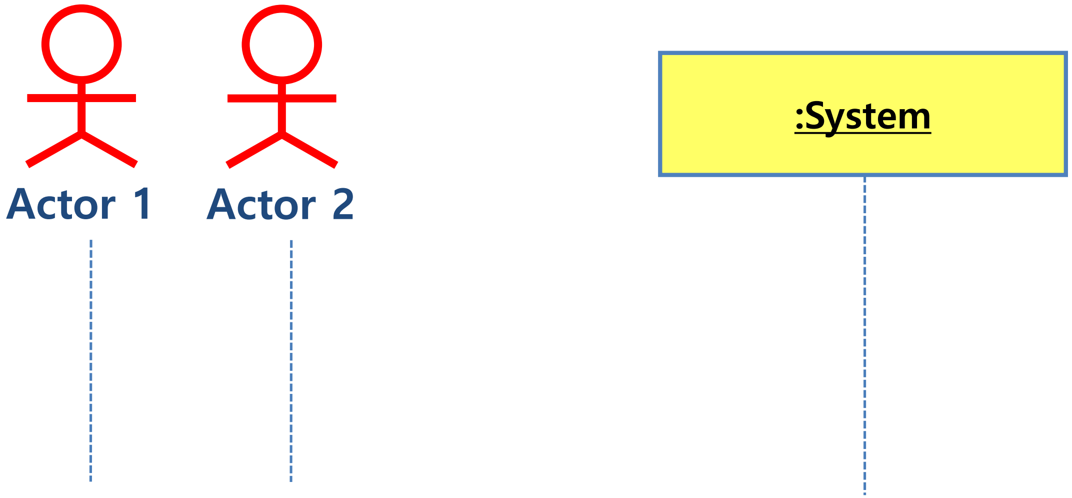
- Description
  - Illustrates events from actors to systems.
  - To investigate the system to build
  - Input : essential use case descriptions, use case diagram
  - Output : A sequence diagram

Activity 2035.
Define System Sequence Diagrams

- What is a system sequence diagram(SSD) ?
  - A picture that shows the events that external actors generate, their orders, and inter-system events
  - All systems are treated as a black box
  - The emphasis of the diagram is events that cross the system boundary from actors to systems
  - SSDs should be defined for
    - Main success scenarios
    - Frequent, complex, or alternative scenarios

Activity 2035.
Define System Sequence Diagrams

- Step
  1. Draw a black box representing the system based on a use case
  2. Identify each actor that directly operate on the system from the typical(normal) course of events in a use case
     - Draw a line for each actor

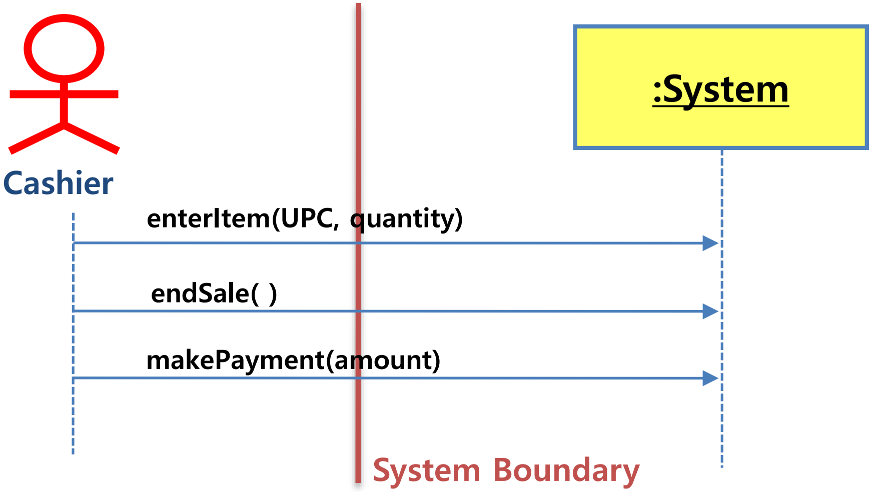


Activity 2035.
Define System Sequence Diagrams

3. Determine system boundary
   - Hardware/software boundary of a device or computer system
   - Department of an organization or Entire organization

- Identify the system(external) events that each actor generates by according to typical course of events in a use case
- Name system events
  - Should be expressed at the level of intent rather than of the physics
  - Name a system event with a verb and an objective like "enterItem"

Scenario: Buy Items



Activity 2035.
Define System Sequence Diagrams

4. Include the use case text which corresponds to system event to the left of the system sequence diagram

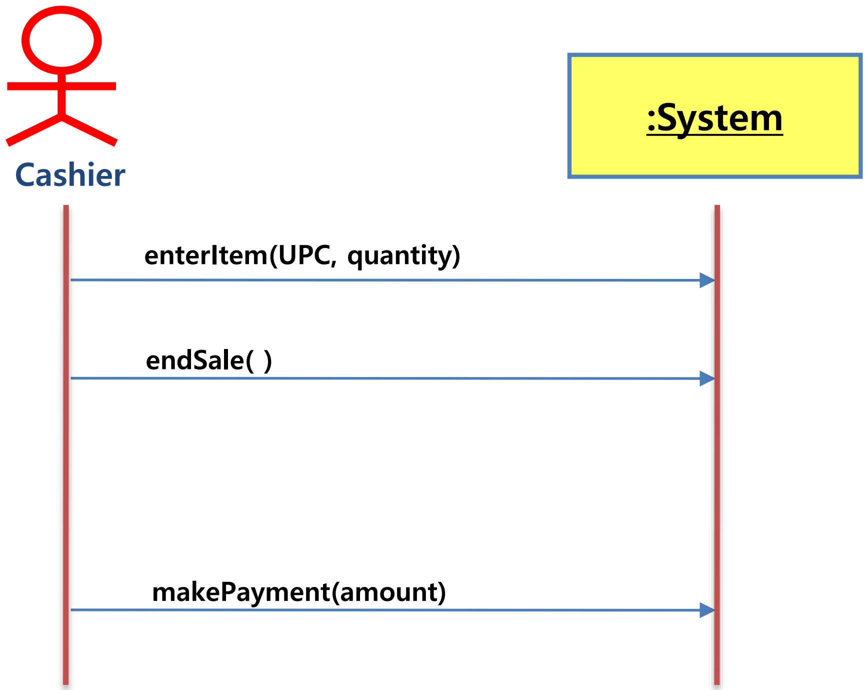
USE CASE: Buy Items

Typical Course of Events

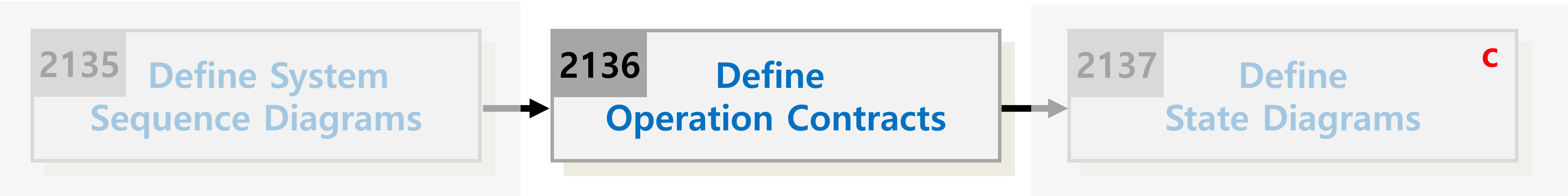
1. This use case begins when a Customer arrives at the POST to checkout with items to purchase.

2. The Cashier records the universal product code(UPC) from each item. If there is more than one of the same item, the Cashier can enter the quantity as well.

3. System determines the item price and adds the item information to the running sales transaction. The description and price of the current item are displayed.



Activity 2036. Define Operation Contracts



2135 Define System Sequence Diagrams

2136 Define Operation Contracts

2137 Define State Diagrams

C

- Description
  - Define contracts for system operations
  - Input : system sequence diagram, conceptual class diagram
  - Output : Operation Contracts

- What is a contract?
  - A document that describes what an operation commits to achieve
  - Written for each system operation to describe its behavior
  - System Operation Contract : Describes changes in states of overall system when a system operation is invoked

Activity 2036. Define Operation Contracts

- Step
  1. Identify system operations from system sequence diagrams
     - A system operation : an operation of the system that executes in response to a system event in sequence diagram.
  2. Fill in operation name sections with contract's names
     - Name: enterItem(upc: number, quantity: integer)
  3. Fill in responsibilities sections
     - Responsibilities: Enter sale of an item and add it to the sale.
       Display the item description and price.
  4. Fill in post-condition sections
     - Post-conditions are declarations about the system state that are true when the operation has finished.
  5. Fill in pre-condition sections
     - Pre-conditions define assumptions about the state of the system at the beginning of the operation.
  6. Fill in other (optional) sections

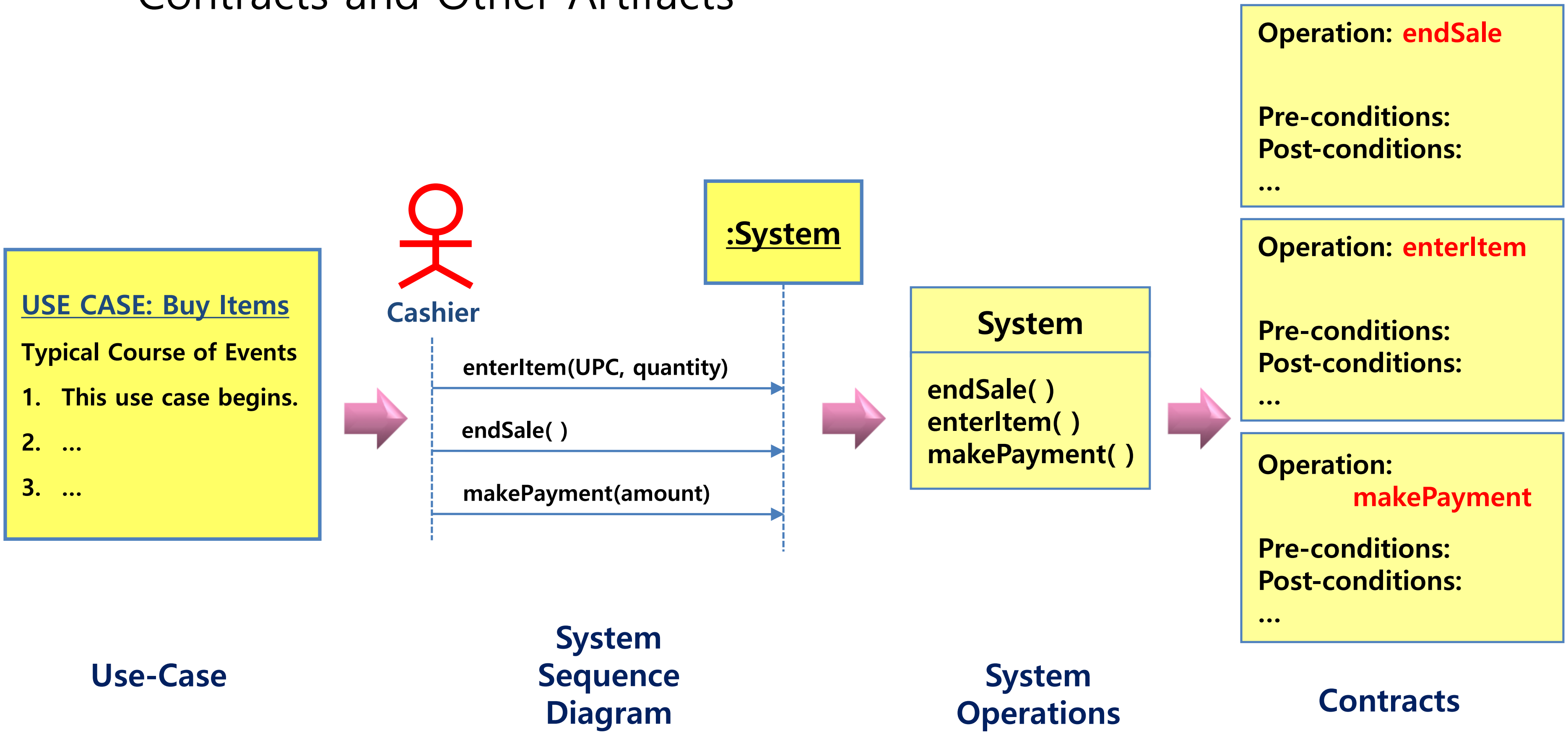
Activity 2036. Define Operation Contracts

- Operation Contracts Format

Name	Name of operation, and parameters
Responsibilities	An informal description of the responsibilities that the operation must fill
Type	Name of type(concept, software class, interface)
Cross References	System function reference numbers, use cases, etc.
Notes	Design notes, algorithms, and so on.
Exceptions	Exceptional cases
Output	Non-UI outputs, such as messages or records that are sent outside of the system
Pre-conditions	Assumptions that the state of the system before execution of the operation
Post-conditions	The state of the system after completion of the operation
…	

Activity 2036. Define Operation Contracts

- Contracts and Other Artifacts



USE CASE: Buy Items

Typical Course of Events

1. This use case begins.
2. …
3. …

Cashier

:System

enterItem(UPC, quantity)

endSale( )

makePayment(amount)

System

endSale( )
enterItem( )
makePayment( )

Operation: endSale

Pre-conditions:
Post-conditions:
…

Operation: enterItem

Pre-conditions:
Post-conditions:
…

Operation: makePayment

Pre-conditions:
Post-conditions:
…

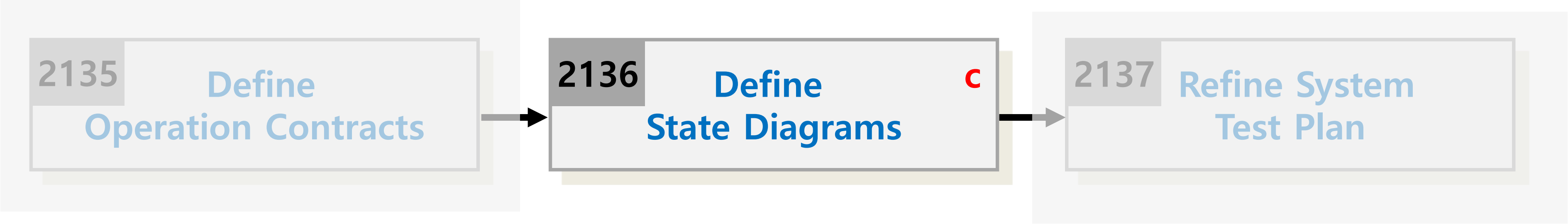
Use-Case

System Sequence Diagram

System Operations

Contracts

Activity 2037. Define State Diagrams



- Description
  - Describes all possible states of the system, use cases, or objects
  - Input : essential use case diagram, conceptual class diagram
  - Output : A state diagrams

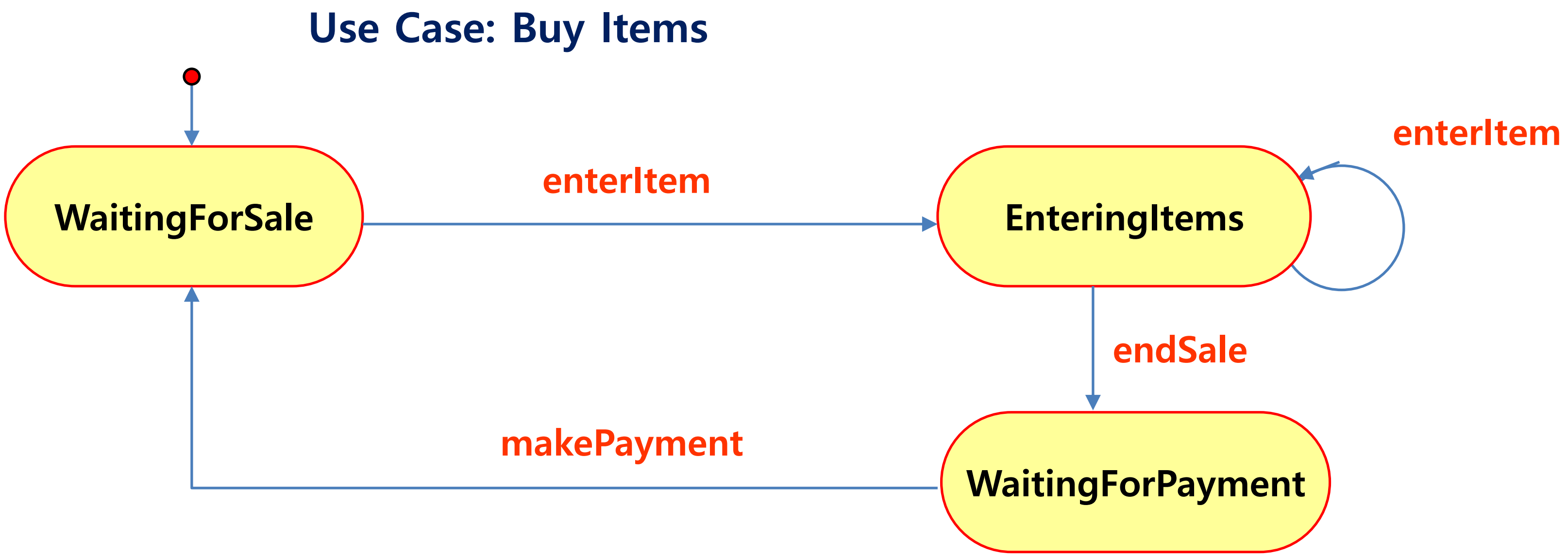
- Three kinds of State diagrams:
  1. Use case state diagram
  2. System state diagram
  3. Class state diagram

Activity 2037. Define State Diagrams

- Event
  - A significant or noteworthy occurrence
  - Ex) a telephone receiver is taken off the hook
- State
  - Condition of an object at a moment in time
  - Ex) a telephone is in the state of being “idle” after the receiver is placed on the hook and until it is taken off the hook
- Transition
  - A relationship between two states that indicates that when an event occurs and the object moves from one state to another
  - Ex) when the event “off hook” occurs, transition occurs from the “idle” to “active” state

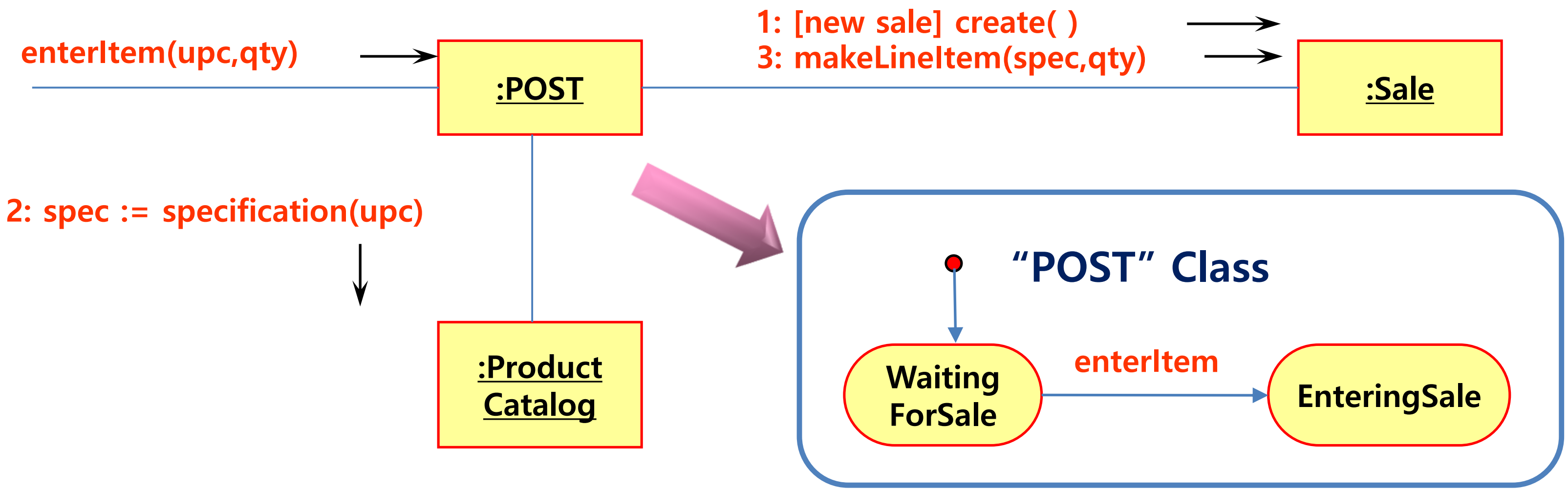
Activity 2037. Define State Diagrams

- Use Case State Diagram
  - A state diagram that depicts the overall system events and their sequence within a use case



Activity 2037. Define State Diagrams

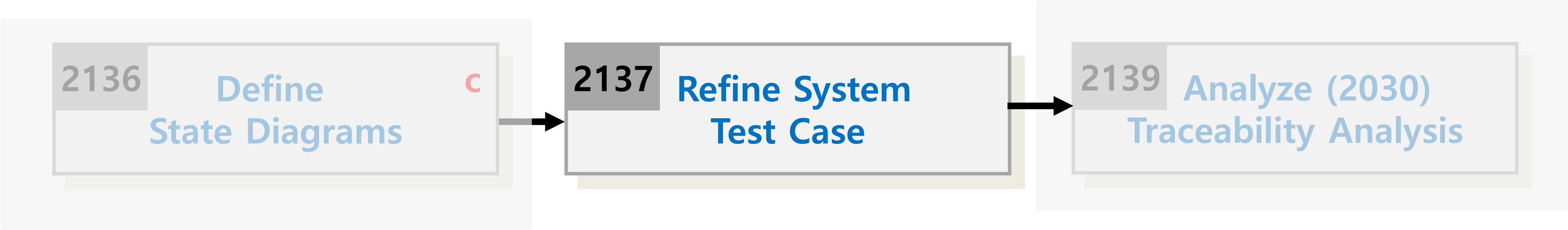
- Class State Diagram
  - A state diagram that depicts state changes of a class across all the use cases
  - Identify a class from interaction diagram
  - A union of all the use case state diagrams



Activity 2037. Define State Diagrams

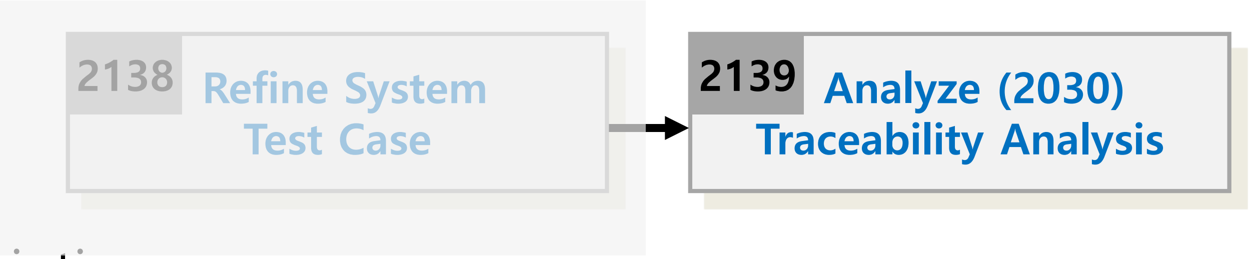
- System State Diagram
  - Identify system events from system sequence diagram
  - Determine sequence of system events
  - Assign system events into transition of state diagram
  - This is an optional activity

Activity 2038. Refine System Test Case



- Description
  - Refine system test plan by using additional information
  - Input : essential use case description, system test plan, sequence diagram
  - Output : refined system test plan
- Step :
  - Refine the results of activity 1009 with the results of analyze process

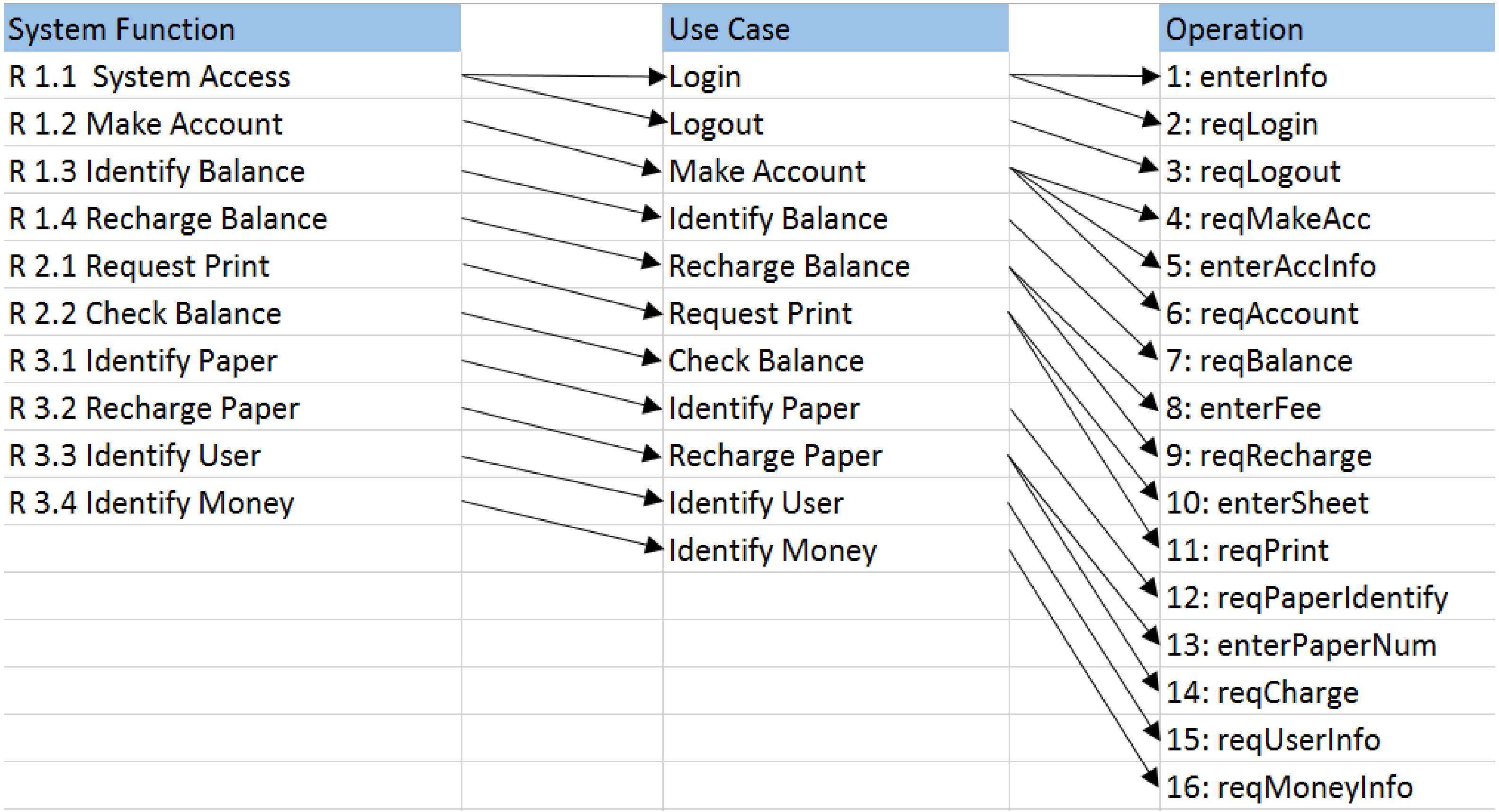
Activity 2039.
Analyze (2030) Traceability Analysis



- Description
  - Analysis the connection of results which are the results of analyze (2030) step
    - Identify the connection of use cases, system sequence diagram and operation contracts
  - Input : Essential use case description, sequence diagram
    operation contracts
  - Output : Traceability analysis result
- Step
  - Writing the relations about the results of each step

Activity 2039.
Analyze (2030) Traceability Analysis

- Example of Analyze traceability



System Function		Use Case		Operation
R 1.1 System Access		Login		1: enterInfo
R 1.2 Make Account		Logout		2: reqLogin
R 1.3 Identify Balance		Make Account		3: reqLogout
R 1.4 Recharge Balance		Identify Balance		4: reqMakeAcc
R 2.1 Request Print		Recharge Balance		5: enterAccInfo
R 2.2 Check Balance		Request Print		6: reqAccount
R 3.1 Identify Paper		Check Balance		7: reqBalance
R 3.2 Recharge Paper		Identify Paper		8: enterFee
R 3.3 Identify User		Recharge Paper		9: reqRecharge
R 3.4 Identify Money		Identify User		10: enterSheet
		Identify Money		11: reqPrint
				12: reqPaperIdentify
				13: enterPaperNum
				14: reqCharge
				15: reqUserInfo
				16: reqMoneyInfo

Summary

- What is the objective of OSP stage 2000?
  - Can you picture the flow of stage 2000?
  - Can you picture the flow of stage 2030 Analyze?
  - Can you clarify the difference between 2030 Analyze and 1000 activities?