KUPE for Parallel System

이철
박소은

Index

Why Parallel Computing?

- Parallel Computing
  - A type of computation in which many calculations or the execution of processes are carried out simultaneously
  - Larger problems can often be divided into smaller ones
- Why parallel computing?
  - Due to the physical constraints preventing frequency scaling
  - Hard to improve performance above the power wall
- Using parallel computing
  - Climate modelling
  - Drug discovery
  - Data analysis

[Reference: An Introduction to Parallel Programming by Peter Pacheco]

Challenges of Parallel Computing

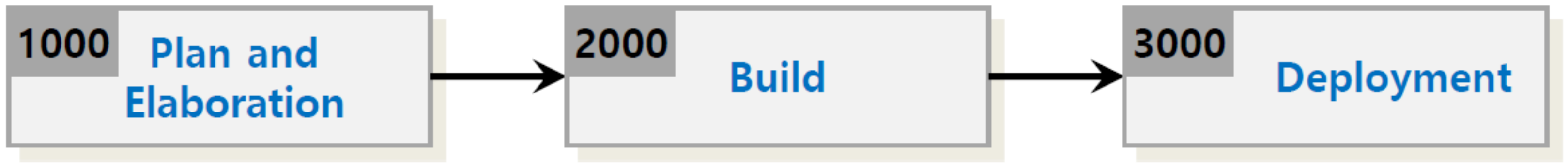
- ➢Synchronization
  - More than two threads read or write the shared resource at the same time
- ➢Communication
  - More than two processes or threads use to share the data that are not in the same memory space of each process or thread
  - For collaboration
  - Message Passing Interface(MPI)
- ➢Load balancing
  - Important that each thread computes the balanced amount of tasks
  - For performance

Solutions

- ➢Synchronization
  - Atomic execution
  - Lock
- ➢Communication
  - Pipe
  - Blocking queue
- ➢Load balancing
  - Task scheduling

KUPE Process

KUPE Process



- Stage 1000: Plan and Elaboration
  - Planning, defining requirements, building prototype, etc
- Stage 2000: Build
  - Construction of the system
- Stage 3000: Deployment
  - Implementation of the system into use

Stage 1000.
Plan and Elaboration

Activity 1003. Define Requirements

- Description
  - Write a requirement specification for a product
  - Input: draft project plan, investigation report
  - Output: a requirement specification

- What is a requirement?
  - A condition or capability needed by a user to solve a problem or achieve an objective
  - 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 document
  - A documented representation of a condition or capabilities as in (1) or (2)

Activity 1003. Define Requirements

➢ Functional Requirements

- 사용자를 계정 별로 관리한다.
  - 계정은 사용자가 원하는 대로 생성하며 사용자 계정은 id/pw와 잔액정보를 가진다.
  - 계정 별로 잔액을 충전 할 수 있다.
  - 사용자는 프린터 사용 전에 계정을 먼저 생성해야 한다.
  - Id/pw 가 일치하는 경우에만 인쇄가 가능하도록 한다.
    - 로그인, 로그아웃이 가능 하다.
  - 여러 명의 사용자가 동시에 요청을 할 수 있다.

Ref. #	Challenge	Function	Category
R1	Synchronization	Check parallelism	Hidden
R2		Allocate sequence	Hidden
R3		Identify sequence	Hidden

Activity 1006. Define Business Use Case

- Description
  - To obtain a deeper understanding of the processes and requirements identified so far
  - Identify business tasks as business use cases, and illustrate their relationships in use case diagrams
  - Input: requirements specification
  - Output: a business use case model (High-level use case)

- Steps
  1. Determine system boundary in order to identify what is external versus internal, and what the responsibilities of the system are
  2. Identify the actors related to a system or organization

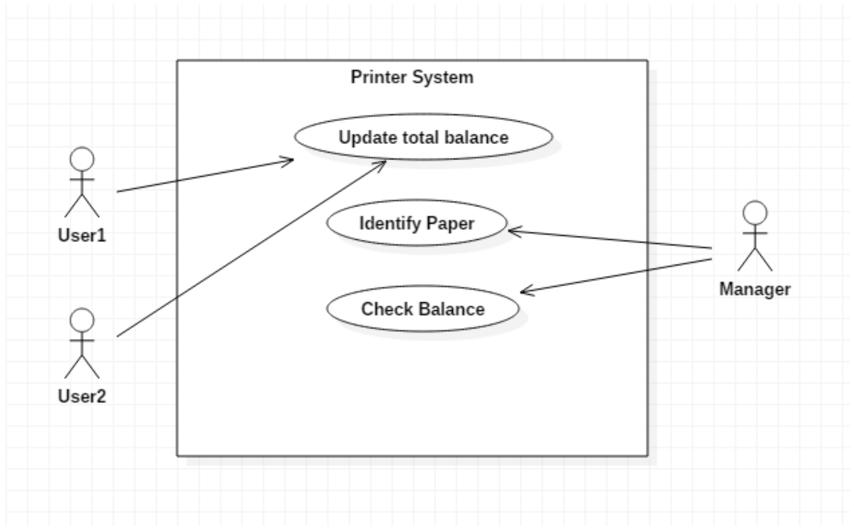
     +) Identify multi-executed functions
  3. Identify user goals for each actor
  4. Record the primary actors and their goals in an actor-goal list

Activity 1006. Define Business Use Case

5. Define use cases that satisfy user goals
6. Allocate system functions identified during the requirements specification into related use cases
7. Categorize identified use cases into primary, secondary, and optional use cases
8. Identify relationships between use cases
9. Draw a use case diagram
10. Describe use cases
11. Rank use cases according to the followings:
    a. Significant impact on the architectural design
    b. Significant information and insight regarding the design
    c. Include risky, time-critical, or complex functions
    d. Involve significant research, or new and risky technology
    e. Represent primary line-of-business processes
    f. Directly support increased revenue or decreased costs

Activity 1006. Define Business Use Case

- Identify multi-accessed functions



Use Case	Update total balance
Actor	Users(multiple)
Description	-이 use case는 전체 수익을 갱신한다. -다수의 사용자가 동시에 접근하는지 확인한다.

Stage 2030.
Analyze

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

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
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

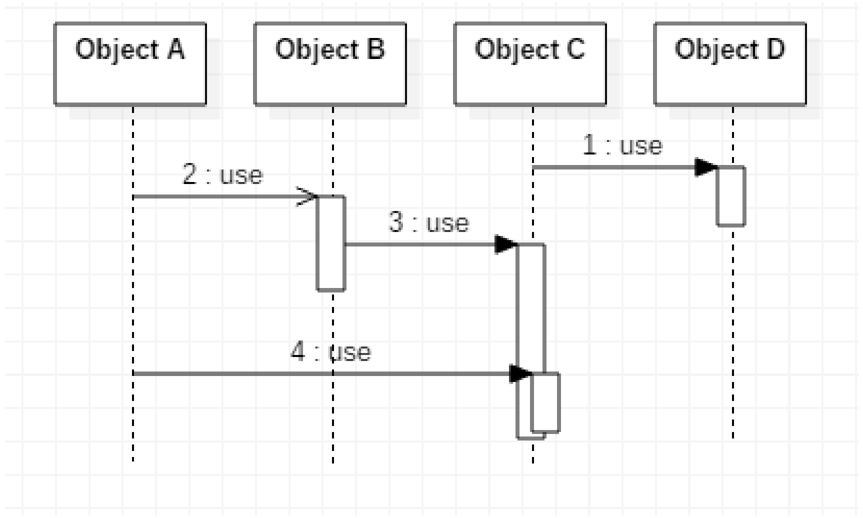
➢Example: “Update total balance”

Use Case	Update total balance
Actor	Users(Multiple)
Purpose	Update total balance
Overview	After a customer requests print, the system withdraw the balance. And then the total balance of the system is updated.
Type	Primary
Cross Reference	…
Pre-Requisites	N/A
Typical Courses of Events	… (S): Identify the simultaneously accessing threads …
Alternative Courses of Events	…
Exceptional Courses of Events	E1: If someone updates the balance already, suspend until the previous update ends

Activity 2030. Define System Data Diagrams

➢Description

- To obtain data usage flow
- Input: Domain Model
- Output: Data diagram



➢Step

1. List objects from domain model
2. Assign usage-relationship between those objects

Activity 2033. 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, system data diagram
- Output: a sequence diagram

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

   +) Discriminate the simultaneously operated part of the program

Activity 2036. Define Operation Contracts

- 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
  2. Fill in operation name sections with contract’s names
  3. Fill in responsibilities sections
  4. Fill in post-condition sections
  5. Fill in pre-condition sections
  6. Fill in other (optional) sections

Activity 2036. Define Operation Contracts

- Operation Contracts

Name	Update total balance
Responsibilities	Update the total balance of the system
Type	System
Cross Reference	…
Notes	…
Exceptions	N/A
Output	N/A
Pre-conditions	Identify the simultaneously accessed objects
Post-conditions	Display the result

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, system data diagram
  - Output: refined system test plan

- Step
  - Refine the results of activity 1009 with the results of analyze process

Stage 2040.
Design

Activity 2044. Define Interaction Diagram

- Description
  - Collaboration diagrams illustrate object interactions in a graph or network format
  - To illustrate how objects interactions via messages to fulfill tasks
  - Input: Real Use Case Descriptions, system data diagram
  - Output: An interaction diagram
  - Standards applied

- Steps
  1. Draw up actors
  2. Deploy objects or classes participating each use case from the real use case descriptions and conceptual class diagram
  3. Design a system of interacting objects to fulfill the tasks
  4. Regard the use case description as a starting point

Activity 2045. Define Design Class Diagram

- Description
  - Describes more details in conceptual class diagram
  - Add navigability, dependency, data type, operation signature, parameters, return types, and so on
  - Input: Interaction Diagram, Conceptual Class Diagram, system data diagram
  - Output: A Design Class Diagram
  - Standards Applied

Stage 2060.
Testing

Activity 2062. Integration Testing

- Description
  - Integration testing is the phase in software testing in which individual software modules are combined and tested as a group

    +) Add synchronization testing
  - Input : Class & Method definitions
  - Output : Integration testing results, reports

Activity 2063. System Testing

- Description
  - System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements
  - Input: Implements results, system test plan and cases
  - Output: System testing results, reports

- Steps
  1. Identify system test cases before defined
  2. Set the test data of test cases for testing
  3. Performing system testing with system test plan and cases

Activity 2063. System Testing

- System test case

Test number	Test item	Description	Use case	System function
…	…	…	…	…
2	Intended result test	Identify the result after multiple threads execute the function		
…	…	…	…	…

Expected Result

- ➢Guarantee the consistency of the parallel program
- ➢Clarify the data dependency between objects
- ➢Consequently, the process can reduce the unexpected result and prevent the malfunction of the parallel system

Q&A