강의노트

09주차 01차시

아키텍처 설계 개요

❖ 학습안내

이번 시간의 학습내용과 학습목표를 확인해보세요.

▣ 학습내용

- 소프트웨어 설계
- 아키텍처 설계
- 아키텍처 모델 결정

▣ 학습목표

- 프로젝트를 수행 시 설계단계를 성공적으로 진행할 수 있다
- 구현된 시스템의 아키텍처를 결정하고 설계에 반영할 수 있다.
- 아키텍처 모델을 결정하기 위하여 참고할 모델을 분석할 수 있다.

❖ 학습내용

[1] 소프트웨어 설계

1. 소프트웨어 설계 정의

◈ 소프트웨어 설계 단계

- 분석된 결과로 목적물에 대한 설계를 수행
- 개념설계와 상세설계를 거쳐 실제 시스템(아키텍처) 설계, 인터페이스 설계, 자료구조(데이터) 설계를 함

◈ 설계단계 주요활동

개발 표준 지침 정의

- 소프트웨어(프로그램) 개발을 위하여 개발표준을 정의

  - 프로그램, 함수, 변수 명 등 명명규칙(Name-rule)
  - 개발 시 유의사항
  - 프로그램 코딩규칙

시스템(아키텍처) 설계

- 개발프로그램, 모듈, 하드웨어, 네트워크, 데이터베이스, 다른 시스템과의 연계 등 전체 구축될 시스템에 대하여 설계

인터페이스 설계

- 시스템, 모듈 간 주고받는 데이터(I/O정의), 함수, 클래스 등의 매개 변수와 같은 인터페이스를 설계

❖ 학습내용

[1] 소프트웨어 설계

1. 소프트웨어 설계 정의(계속)

◈ 설계단계 주요활동(계속)

자료구조(DB) 설계

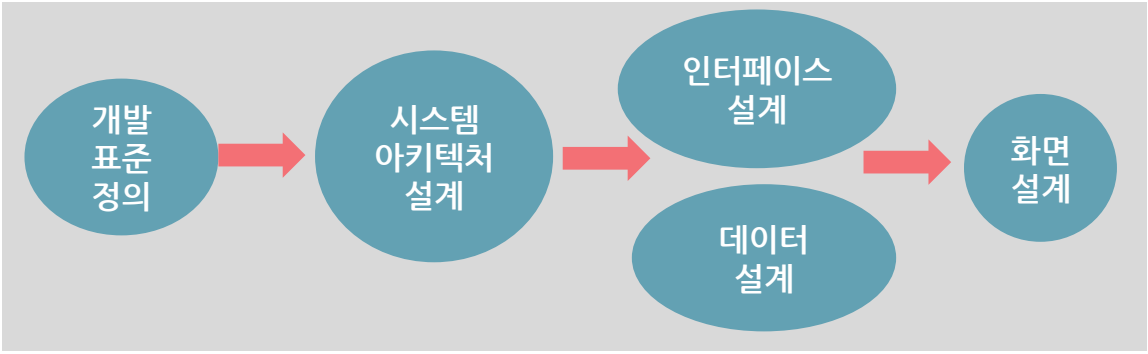
- 데이터베이스 설계와 관련되어 ERD, CRUD(Create/Read/Update/Delete)정의 등을 설계

사용자 화면 설계

- 실제 정보시스템에서 사용자가 직접 조작하는 부분인 화면에 대하여 개발자가 구현할 수 있는 수준으로 설계
- 인터페이스 설계에 포함하는 개념(UI: User Interface)으로도 다룰 수 있음

2. 소프트웨어 설계 절차

◈ 소프트웨어 설계 절차 개요



- 소프트웨어 설계는 다음과 같은 순서로 진행
- 시스템(아키텍처) 설계 부분은 다른 부분을 설계 하는데 있어서 뼈대 역할을 함

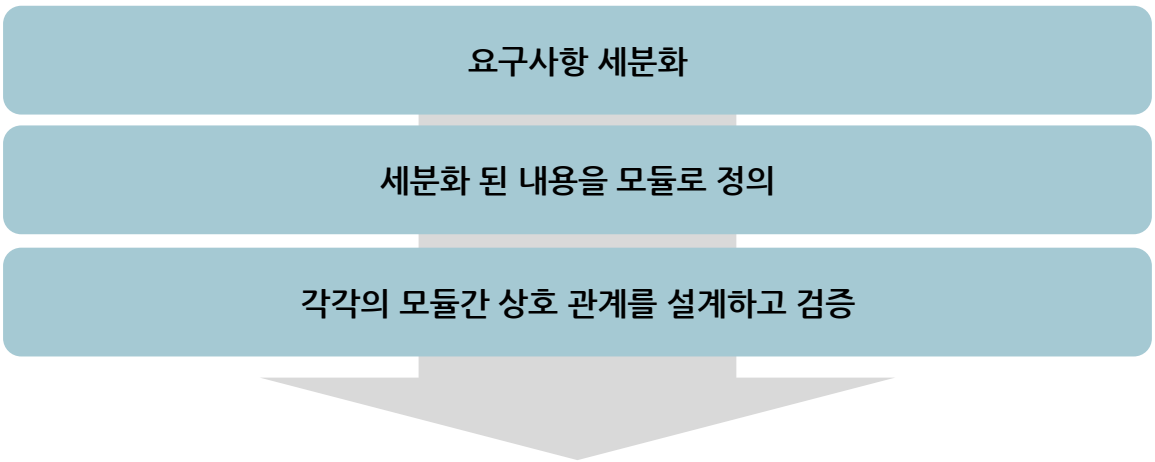
❖ 학습내용

[1] 소프트웨어 설계

2. 소프트웨어 설계 절차(계속)

◈ 아키텍처 설계 절차

◈ 전체 시스템을 Top-down방식으로 분석



- 모듈을 정의할 때 적합한 아키텍처 스타일을 정하고 이에 적합하도록 모듈을 정의

◈ 데이터 설계 절차

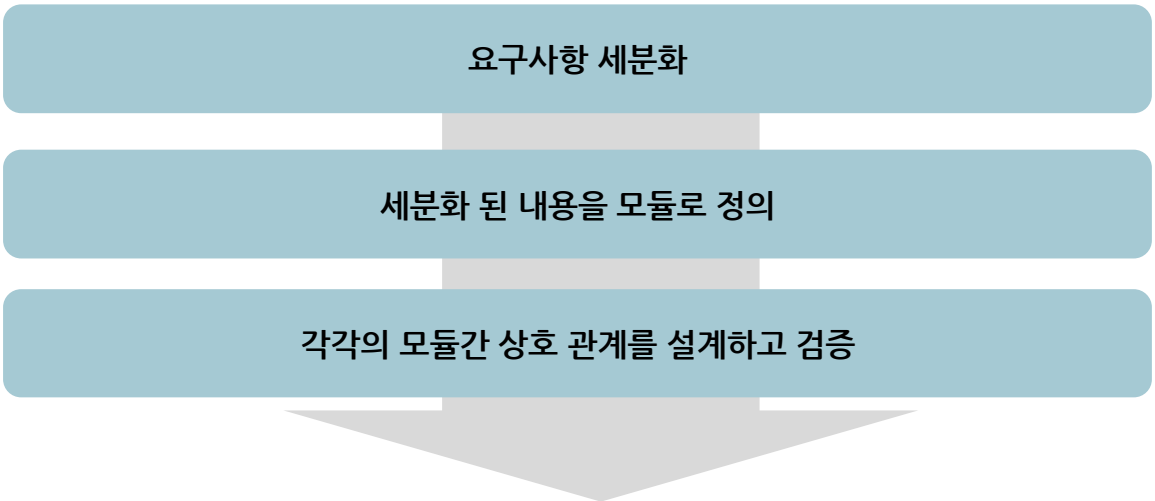
- 정보시스템에서 사용할 데이터를 추상화함
- 데이터와 데이터 구조(데이터 스키마)를 정의
- 외부 인터페이스 설계 절차
- 외부 인터페이스 선정 및 목록화
- 외부 인터페이스 정의
- 시스템 인터페이스 설계 절차
- 시스템 인터페이스 목록화
- 시스템 인터페이스 선정 및 정의

❖ 학습내용

[1] 소프트웨어 설계

3. 아키텍처 설계 절차

◈ 설계 절차



3. 1. 요구사항 세분화

◈ 요구사항 명세서의 검토

① 시스템이 수행할 모든 기능과 시스템에 영향을 미치는 제약 조건이 명확하게 기술되었는지 검토

② 명세 내용은 사용자와 개발자 모두가 이해하기 쉽고 간결하게 작성되었는지 검토

③ 기술된 모든 요구사항은 검증이 가능하므로 원하는 시스템의 품질, 상대적 중요도, 품질의 측정, 검증 방법 및 기준이 명시되었는지 검토

④ 요구사항 명세서는 시스템의 외부 행위를 기술하는 것으로, 특정한 구조나 알고리즘을 사용하여 설계하지 않게 되었는지 검토

⑤ 참여자들이 시스템의 기능을 이해하거나, 변경에 대한 영향 분석 등을 위하여 계층적으로 구성되었는지 검토

⑥ 요구사항을 쉽게 참조할 수 있도록 고유의 식별자를 가지고 번호화하고, 모든 요구사항이 동등한 것이 아니므로 요구사항을 우선 순위화하였는지 검토

⑦ 기능 및 비기능 요구사항이 이해관계자별로 구분되었는지 검토

❖ 학습내용

[1] 소프트웨어 설계

3. 1. 요구사항 세분화(계속)

◈ 요구사항 명세서의 검토(계속)

시스템SW 요구사항 정의서에 액터별로 기능 요구사항이 반영되어 있는지

시스템SW 요구사항에 비기능 요구사항이 모두 반영되어 있는지

제약사항이 모두 기술되어 있는지

- 시스템SW 아키텍처 제약사항이 모두 있는지 검토
- 기능 요구사항 이외에 필요한 비기능 요구사항이 모두 들어있는지 반드시 검토

◈ 이해관계자별 기능 및 비기능 요구사항을 확인

이해관계자를 참고하여 이해관계자별 시스템SW 엔지니어링의 주요 핵심이 되는 기능 요구사항의 내용이 무엇인지 확인

이해관계자별 품질 관련 비기능 요구사항의 내용이 무엇인지 확인

◈ 기능 요구사항을 세분화

1 세분화 기준 선정

- 핵심 기능 요구사항의 기준을 정함
- 전체 모듈을 고려한 선정기준을 정함

2 기존 시스템 SW 분석

- 개발되었던 시스템 SW 아키텍처의 특성을 고려
- 이미 제품으로 나와 있는 시스템 SW의 특성을 참고

❖ 학습내용

[1] 소프트웨어 설계

3. 1. 요구사항 세분화(계속)

◈ 기능 요구사항을 세분화 (계속)

3 선정기준을 고려하여 기능 요구사항 세분화

4 제약사항을 고려하여 기능 요구사항 세분화

- 연계할 시스템과 재사용할 SW 등의 인프라 여부를 확인

◈ 비기능 요구사항을 세분화

1 세분화 기준 선정

- 이해관계자별 비기능 요구사항 세분화 기준 선정
- 품질 속성을 고려한 요구사항 세분화 기준 선정

2 기존 시스템 SW 분석

- 기존 시스템 SW 제품들의 성능 요구사항을 검토
- 현재 시스템 SW의 요구사항이 있는지 검토

3 선정기준을 고려하여 기능 요구사항 세분화

4 제약사항을 고려하여 기능 요구사항 세분화

예
- 중요도를 고려
- 우선순위를 고려

❖ 학습내용

[1] 소프트웨어 설계

3. 1. 요구사항 세분화(계속)

◈ 세분화된 기능 요구사항을 분류

- 세분화된 기능 요구사항을 상호 독립적인 세부 단위로 분류

세분화된 기능 요구사항의
상호 독립성 여부 확인

세분화된 기능 요구사항을
기준별 기능 요구사항으로 정리

- 세분화된 기능 요구사항을 기준별 기능 요구사항으로 정리

모듈별 기능 요구사항으로 분류	인터페이스별 기능 요구사항으로 분류	기타 기능 요구사항으로 분류
모듈별 기능 요구사항으로 분류	인터페이스별 기능 요구사항으로 분류	기타 기능 요구사항으로 분류

- 아키텍처 설계 시 반영하는 기타 기능 요구사항에는 다음과 같은 것이 있을 수 있음
- 배포 방식: 서버를 통한 배포, 일괄 배포, 온·오프라인을 통한 배포 등
- 업그레이드 방식: 서버를 통한 버전업, 일괄 버전업, 온·오프라인 버전업 등
- 요금 결제 방식: 제품구매 시 결제, 라이센스 구매

❖ 학습내용

[1] 소프트웨어 설계

3. 1. 요구사항 세분화(계속)

◈ 세분화된 비기능 요구사항에 대해서 분류 및 검토

- 세분화된 비기능 요구사항에 대해서 상호 독립적인 세부 단위로 분류 및 검토

세분화된 비기능 요구사항의 상호 독립성 여부 확인

세분화된 기능 요구사항을 기준별 기능 요구사항으로 정리

- 품질 속성별 비기능 요구사항으로 분류
- 품질 속성별 품질 시나리오를 작성
- 품질 시나리오를 문서화 하고 구조화

검토하여 중복된 내용 삭제

3. 2. 세분화 된 내용을 모듈로 정의

◈ 모듈 정의

- 분류된 기능 및 비기능 요구사항의 핵심 기능을 고려하여 모듈 범위 정의

핵심 기능 및 비기능 요구사항을 고려한 핵심 모듈 블록 정의

핵심 모듈 사이의 관계를 고려하여 관계 정의

❖ 학습내용

[1] 소프트웨어 설계

3. 2. 세분화 된 내용을 모듈로 정의(계속)

◈ 모듈 정의(계속)

- 아키텍처 스타일을 고려하여 모듈을 정의

1	적합한 아키텍처 스타일을 정하고 전체적인 아키텍처의 논리적 구조 정의
2	핵심 비기능 요구사항을 고려하여 기본 아키텍처 구조 정의
3	아키텍처 설계에 재사용하는 기존 소프트웨어나 모듈이 있는지 확인

- 아키텍처 스타일은 이번 강의 및 다음 강의에서 심도있게 다룸
- 이해 당사자 관점에서 정의
- 모듈 정의 시 내용이 중복되는지 확인하고, 중복성 및 재사용성을 충분히 고려하여 모듈을 정의

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증

◈ UML을 이용하여 모듈 상호 관계를 설계

- UML다이어그램 중 클래스 다이어그램과 컴포넌트 다이어그램을 작성하여 모듈간 상호 관계를 설계함

❖ 학습내용

[1] 소프트웨어 설계

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증(계속)

◈ UML을 이용하여 모듈 상호 관계를 설계(계속)

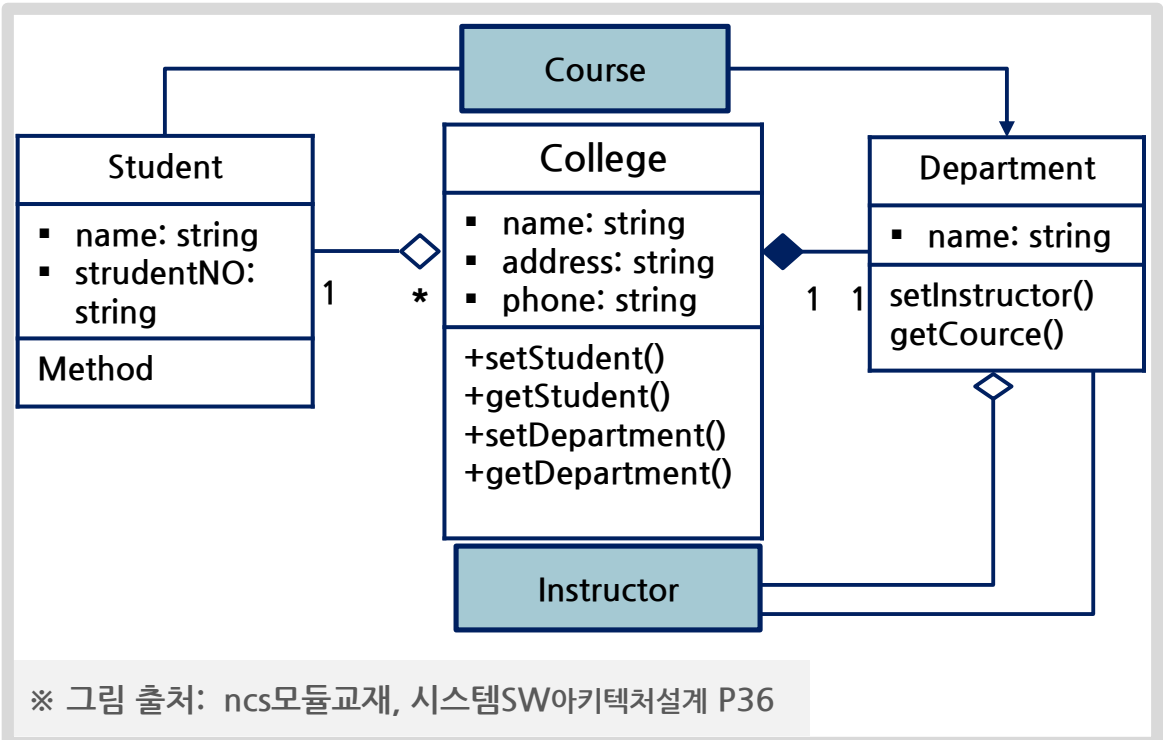
클래스 다이어그램 작성 ➡ 클래스(Class)를 정의

- 클래스는 이름, 속성, 메소드 3개로 구분
- 먼저 박스를 그리고 박스 위에 클래스 이름을 작성
- 일반적으로 추상 클래스의 경우 이탤릭체로 나타내고 인터페이스 클래스는 <<interface>>를 추가
- 속성은 객체가 가지는 모든 필드를 포함
- 메소드에는 가장 일반적인 메소드 및 상속된 메소드는 포함되지 않음

클래스 다이어그램 작성 ➡ 클래스 사이의 관계를 표시

- 클래스 사이의 관계는 일반화 관계, 실체화 관계, 연관 관계, 의존 관계로 분류, 이를 다이어그램으로 표시

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증- 클래스 다이어그램 예시



※ 그림 출처: ncs모듈교재, 시스템SW아키텍처설계 P36

❖ 학습내용

[1] 소프트웨어 설계

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증(계속)

◈ UML을 이용하여 모듈 상호 관계를 설계(계속)

컴포넌트 다이어그램 작성 → 컴포넌트를 정의

- 컴포넌트로 주로 표현되는 개념은 소프트웨어 컴포넌트, 실행파일, 라이브러리, 파일 등을 포함

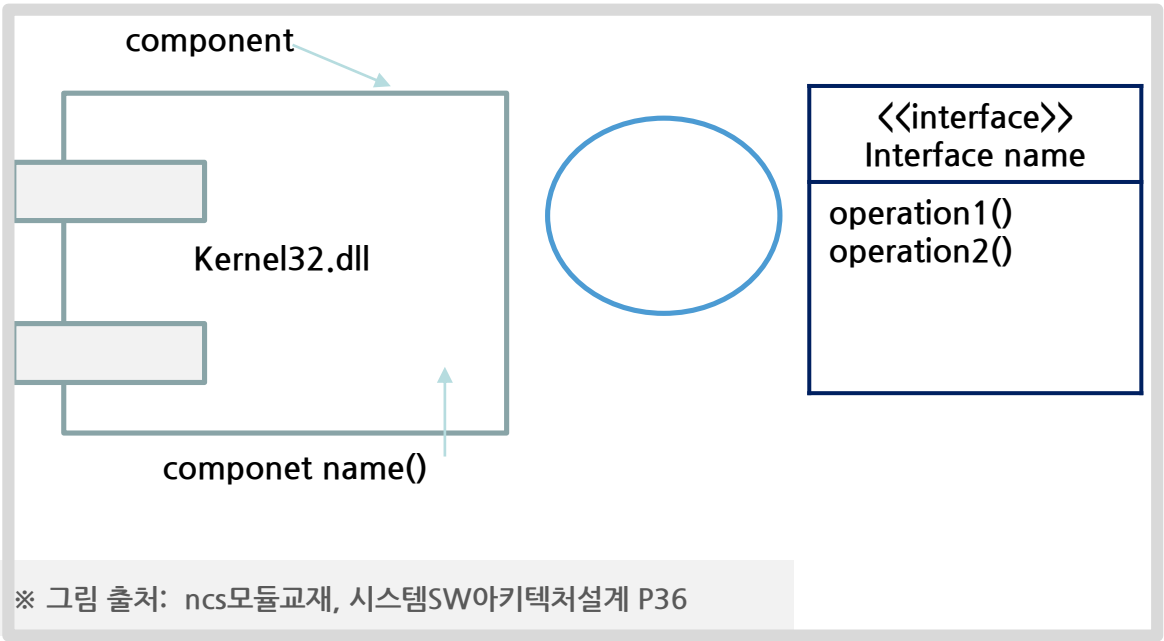
컴포넌트 다이어그램 작성 → 컴포넌트 사이의 관계를 표현

1 인터페이스(Interface) 관계를 표현

- 클래스, 컴포넌트, 서브시스템 등 요소들이 제공하고 있는 서비스를 명세화하기 위해 사용하는 요소로 인터페이스를 표현

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증- 컴포넌트 다이어그램 예시

◈ 컴포넌트와 인터페이스의 표현



※ 그림 출처: ncs모듈교재, 시스템SW아키텍처설계 P36

❖ 학습내용

[1] 소프트웨어 설계

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증(계속)

◈ UML을 이용하여 모듈 상호 관계를 설계(계속)

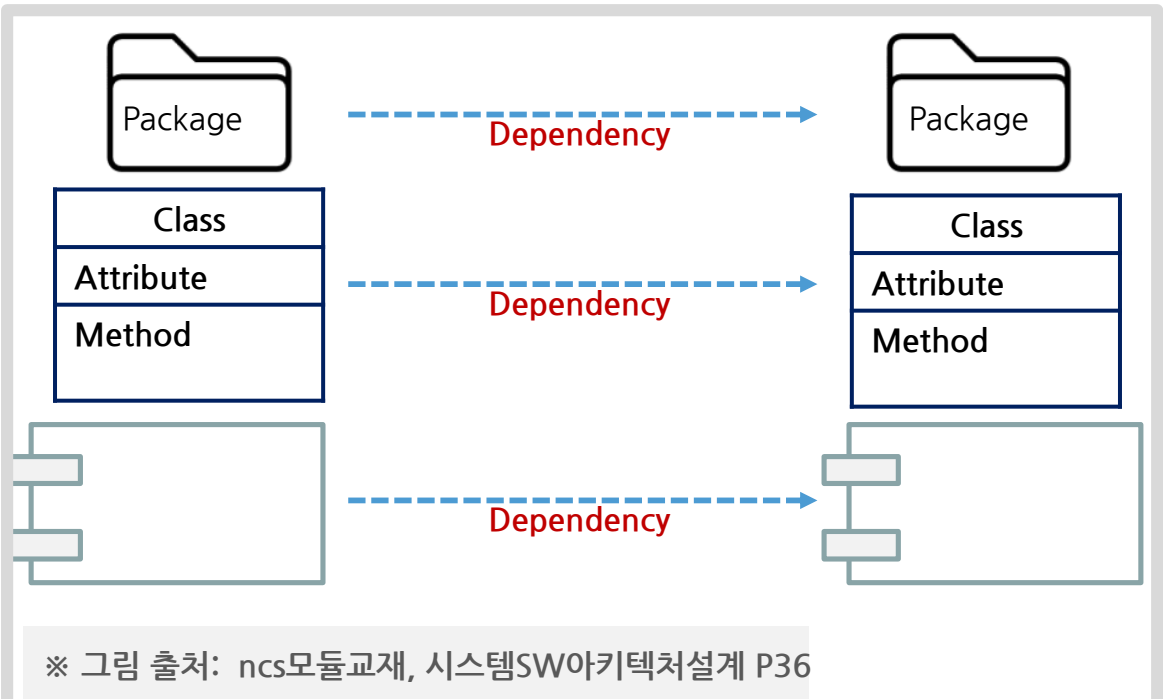
컴포넌트 다이어그램 작성 ➡ 컴포넌트 사이의 관계를 표현

2 의존(Dependency) 관계 표현

- 두 요소 사이의 의미적 관계를 의미
- 하나의 요소의 변경이 다른 요소에 영향을 주는 관계로 패키지, 클래스, 컴포넌트 사이의 의존에 주로 사용

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증- 컴포넌트 다이어그램 예시(계속)

◈ 의존 관계 표현



※ 그림 출처: ncs모듈교재, 시스템SW아키텍처설계 P36

❖ 학습내용

[1] 소프트웨어 설계

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증(계속)

◈ UML을 이용하여 모듈 상호 관계를 설계(계속)

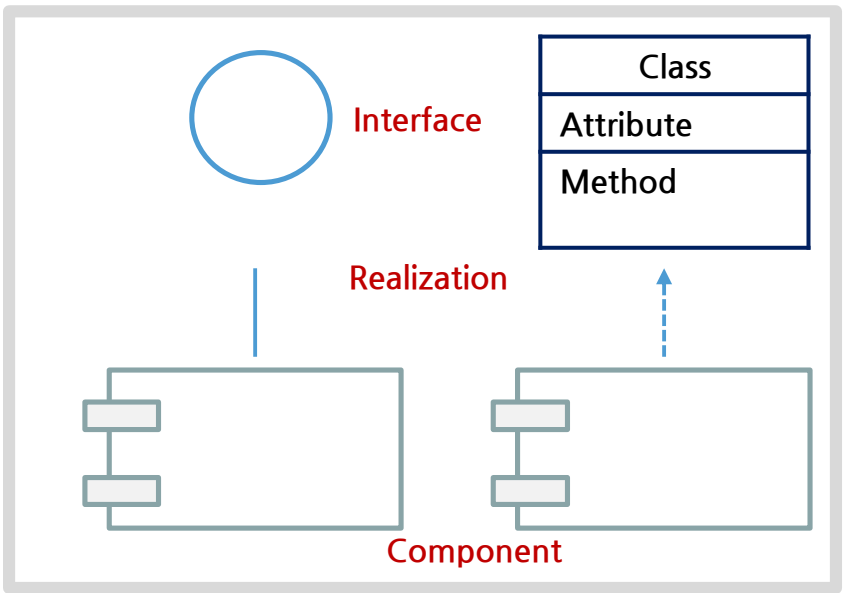
컴포넌트 다이어그램 작성 ➡ 컴포넌트 사이의 관계를 표현

3 실체화(Realization) 관계 표현

- 명세(Specification)와 구현 요소 간의 관계를 의미
- 명세에서 정의되고 구현에서 실체화되는 관계
- 실체화는 협동, 클래스, 서브시스템, 컴포넌트 등에서 사용

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증- 컴포넌트 다이어그램 예시(계속)

◈ 실체화 관계의 표현 및 전체 컴포넌트 다이어그램

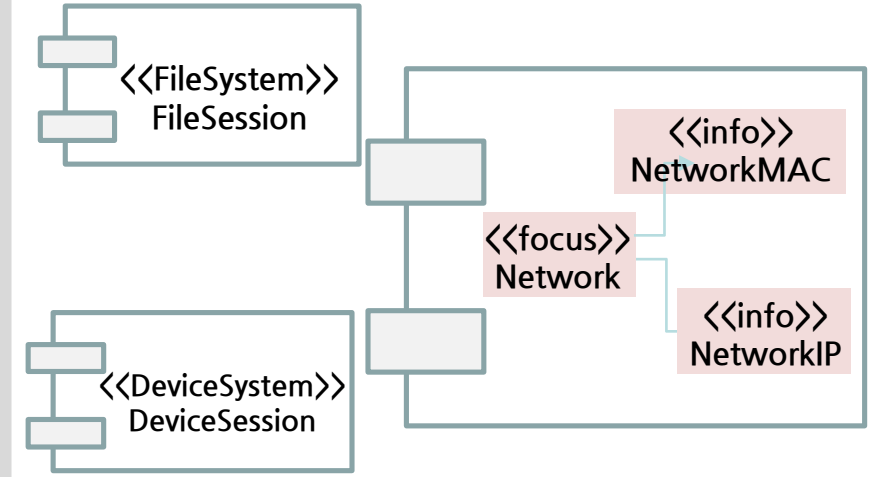


❖ 학습내용

[1] 소프트웨어 설계

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증- 컴포넌트 다이어그램 예시(계속)

◈ 실체화 관계의 표현 및 전체 컴포넌트 다이어그램



※ 그림 출처: ncs모듈교재, 시스템SW아키텍처설계 P37

3. 3. 각각의 모듈간 상호 관계를 설계하고 검증(계속)

◈ 자료 흐름도를 이용하여 모듈 상호 관계를 설계

◈ 자료 흐름도(DFD, Data Flow Diagram)

① 요구사항 분석을 통하여 프로세스, 데이터 저장, 외부 엔티티 등을 찾아서 정의

② 각각의 프로세스에 입력하는 데이터와 처리 내용, 출력 데이터 등 작성

③ 각 프로세스를 배치하고, 이들 사이의 데이터 흐름을 화살표로 표시

④ 모든 데이터 흐름에 이름 부여

⑤ 데이터 흐름도의 규모가 클 경우, 여러 개의 서브 프로세스를 하나의 프로세스로 묶어 표현

⑥ 상위 레벨 DFD를 먼저 작성하고 여러 개의 서브 프로세스로 묶은 프로세스마다 데이터흐름도 표현

⑦ 서브 프로세스의 규모가 크다면 이를 다시 분할하여 작성하고 전체 레벨은 3레벨 정도로 작성

❖ 학습내용

[2] 아키텍처 설계

1. 아키텍처 설계 개념

◈ 아키텍처 설계 정의

아키텍처 설계(Architectural Design)

아키텍처는 각각의 서비스를 제공하는 서브시스템과 이 서브시스템들을 제어하고 통신을 위하여 어떠한 일들을 하여야 하는 것

→ 설계 프로세스의 결과물을 소프트웨어 아키텍처라 함

> "요구분석 이후 구현을 위한 설계의 단계에서 첫 번째 수행은 아키텍처를 설계하는 것"

◈ 아키텍처를 명세화(설계) 해야 하는 이유

✓ 관련자와의 의사소통 도구

- 시스템 관련자(System Stakeholders)와 의견을 교환하는데 사용함

✓ 시스템을 분석하는 도구

- 요건 등이 적용 가능한지 시스템을 분석하는데 사용함

✓ 대단위 시스템에서 재사용 활용

- 대단위 시스템에서 중복 구현 없이 서브시스템이나 모듈 등을 재사용하기 위하여 판단하는데 유용함

❖ 학습내용

[2] 아키텍처 설계

2. 아키텍처 설계 고려사항

◈ 5가지 고려사항

1 성능(Performance)

- 중요한 오퍼레이션은 내부에서 처리하고, 통신은 최소화함
- 잘게 쪼개진 컴포넌트보다는 보다 큰 구성요소로 성능을 높임

2 보안성(Security)

- 내부 계층 안에 주요 자산들을 위치시키는 계층적 아키텍처를 사용

3 안전성(Safety)

- 안전이 필요한 중요기능은 내부에 위치시키며, 서브시스템은 작게 유지함

4 가용성(Availability)

- 장애 등에 대비하기 위하여 컴포넌트나 메커니즘의 중복적 구현이 되어 있어야 함

5 유지보수성(Maintainability)

- 세분화 되어 교체 가능한 컴포넌트를 사용하여야 함

❖ 학습내용

[2] 아키텍처 설계

3. 아키텍처 설계 결정

◈ 아키텍처 설계 결정을 위하여 고려하여야 할 사항

1	참고아키텍처	설계중인 시스템을 위한 템플릿으로서 작동할 수 있는 일반 응용 시스템의 아키텍처가 존재하는가?
2	분산프로세스	시스템이 다수의 프로세서로 어떻게 분산될 것인가?
3	아키텍처 스타일	시스템에 적절한 아키텍처 스타일은 무엇인가?
4	시스템구성 접근법	시스템을 구성하는데 이용되는 기본적인 접근법은 무엇이 될 것인가?
5	모듈분해	시스템에서 구성 단위가 모듈들로 어떻게 분해될 것인가?
6	제어	시스템에서 구성 단위들의 오퍼레이션을 제어하기 위해 어떤 전략이 사용될 것인가?
7	설계평가	아키텍처의 설계를 어떻게 평가할 것인가?
8	문서화	시스템의 아키텍처를 어떻게 문서화할 것인가?

아키텍처 설계 시 고려하여야 할 사항을 참고하여
아키텍처를 설계하고 설계물을 결정한 후 구현에 나서게 됨

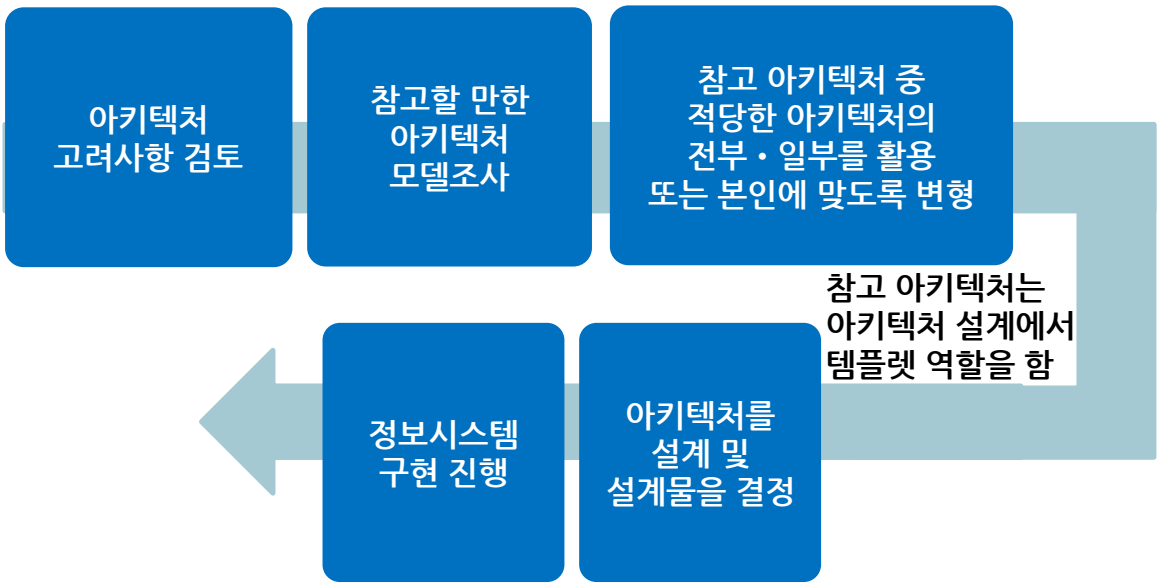
가장 중요한 일은 참고할 만한 아키텍처 모델을 조사한 후
전부 또는 일부분을 템플릿으로 활용하고 본인에 맞게 변형하여
아키텍처를 설계해야 함

❖ 학습내용

[3] 아키텍처 모델 결정

1. 참고 아키텍처 모델

◈ 아키텍처 설계 진행 순서



◈ 아키텍처 모델 결정

- 아키텍처 모델을 결정하기 위하여 기존 구성된 아키텍처 모델을 많이 참고하고 있음
- 아키텍처 모델을 구분하는 방식으로는 하드웨어 시스템 구성방법에 따라 분류하거나 응용(어플리케이션) 차원에서 분류하는 방법이 있음
- 시스템 구성방법에 따라 분류하는 경우는 다음의 세가지가 대표적

저장소 모델 (Repository Model)	Client/Server 모델	계층적 모델 (Layered Model)

❖ 학습내용

[3] 아키텍처 모델 결정

2. 참고 일반 아키텍처 모델 1

◈ 저장소 모델(Repository Model)

- 중앙에 집중된 공유 데이터 베이스를 기반으로 하는 시스템 모델

예 Mainframe 체계의 시스템

장점

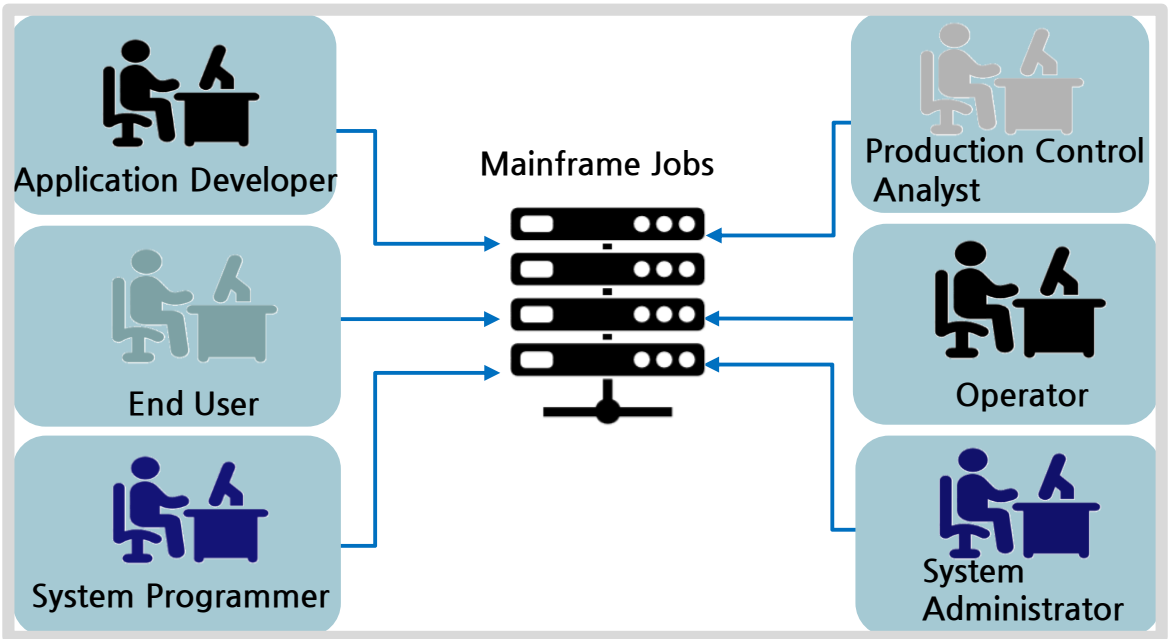
- 다량의 데이터를 공유하는데 효과적
- 백업, 보안, 접근제어, 오류 등이 중앙 집중적으로 통제가능

단점

- 데이터가 중앙 집중되어 있기 때문에 임시적으로 중요치 않은 데이터도 중앙에 모여 저장되어야 함
- 시스템의 유연성이 떨어지기 때문에 한 단위모듈을 수정 보안하는 작업도 전체 시스템의 영향을 줄 수 있음

2. 참고 일반 아키텍처 모델 1 - 저장소 모델 예시

◈ 중앙집중형 메인프레임 구조 사례

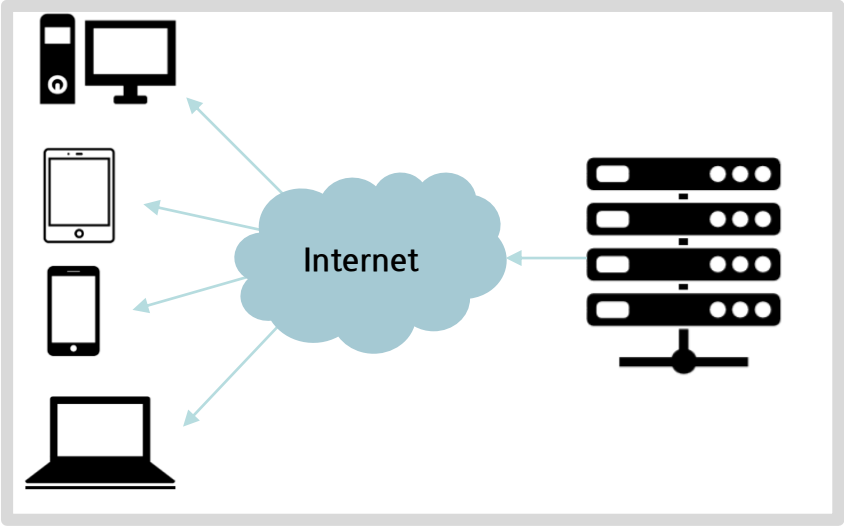


❖ 학습내용

[3] 아키텍처 모델 결정

3. 참고 일반 아키텍처 모델 2

◈ Client/Server 모델



- 서비스와 서버, 클라이언트의 집합으로 구성되는 시스템 모델
- 일반적인 웹 시스템 등이 여기에 해당
- 분산아키텍처로 분산된 프로세스와 네트워크의 이점을 이용할 수 있으나, 데이터의 중복과 불일치를 처리해야 하는 복잡성도 있음

◈ 계층적 모델(Layered Model)

- 시스템을 계층적으로 구성하는 시스템 모델

예 형상관리 시스템 계층 - 객체관리 시스템 계층 - 데이터베이스 시스템 계층 - 운영체계 계층으로 시스템을 나누어 보는 체계

장점

- 계층적 모델은 시스템을 점증적으로 개발할 수 있음

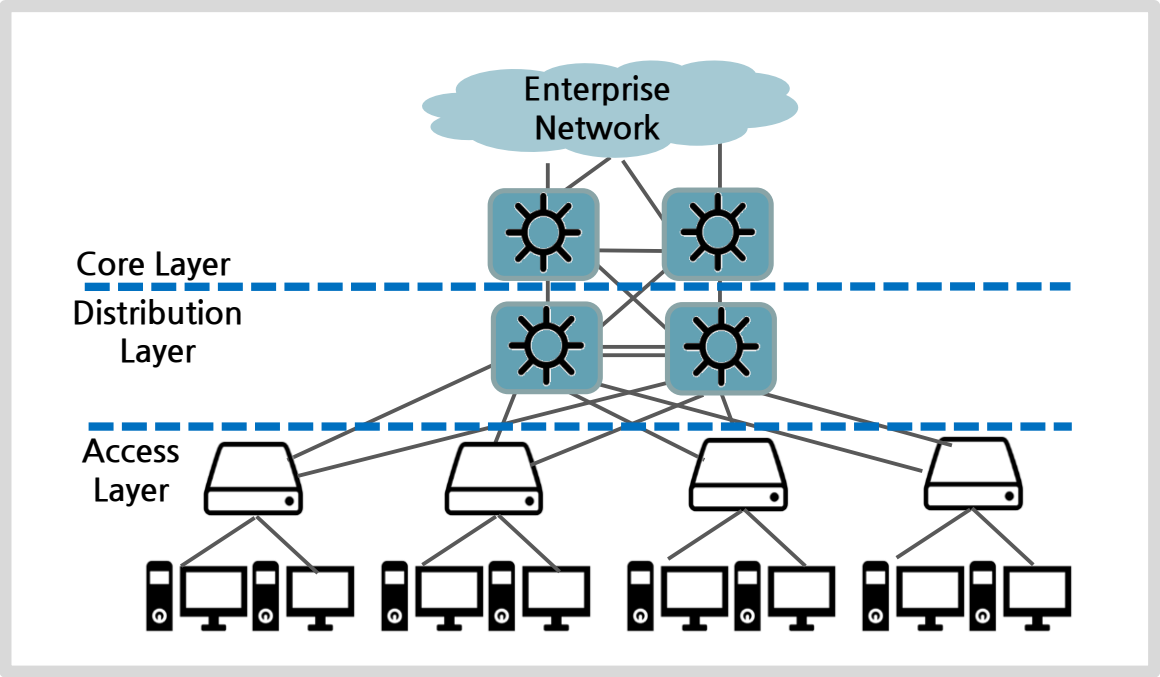
단점

- 시스템을 정확히 어떤 계층으로 확실히 분리할 수 있는지, 모든 계층들을 합하였을 때 전체가 되는지를 고려해야 하는 문제가 있음

❖ 학습내용

[3] 아키텍처 모델 결정

3. 참고 일반 아키텍처 모델 2 - 계층적 모델 예시



❖ 핵심정리

1. 소프트웨어 설계

- 소프트웨어 설계 단계는 개념설계와 상세설계를 거쳐 실제 시스템(아키텍처) 설계, 인터페이스 설계, 자료구조(데이터) 설계를 함
- 소프트웨어 설계단계에서는 개발 표준 지침 정의, 시스템(아키텍처)설계, 인터페이스 설계, 자료구조(데이터) 설계, 사용자 화면 설계가 진행됨
- 아키텍처 설계절차는 요구사항 세분화, 세분화된 내용을 모듈로 정의, 각각의 모듈간 상호관계를 설계하고 검증하는 절차임

2. 아키텍처 설계

- 아키텍처는 각각의 서비스를 제공하는 서브시스템과 이 서브시스템들을 제어하고 통신을 위하여 어떠한 일들을 하여야 하는 것으로 설계 프로세스의 결과물을 소프트웨어 아키텍처라 함

3. 아키텍처 모델 결정

- 아키텍처 설계 진행 순서는 아키텍처 고려사항 검토, 참고할 만한 아키텍처모델을 조사, 참고 아키텍처 중 적당한 아키텍처의 전부 또는 일부를 활용 또는 본인에 맞도록 변형, 아키텍처를 설계 및 설계물을 결정, 정보시스템 구현 진행의 절차로 진행됨