1 아키텍쳐와 소프트웨어 아키텍쳐

01 아키텍쳐와 소프트웨어 아키텍쳐

1 아키텍쳐의 정의

- 특정 사물의 특성을 결정짓는 기본 구조
- 구성요소 사이를 어떻게 연관시키는 가에 대한 설계

01 아키텍쳐와 소프트웨어 아키텍쳐

2 소프트웨어 아키텍쳐의 정의

외부에서 인식할 수 있는 특성이 담긴 소프트웨어의 골격이 되는 기본 구조로서 다음을 포함함

- 구성 요소
- 구성 요소들 사이의 관계
- 구성 요소들이 외부에 드러내는 속성
- 구성 요소들과 주변 환경 사이의 관계
- 구성 요소들이 제공하는 인터페이스
- 구성 요소들의 협력 및 조립 방법

01 아키텍쳐와 소프트웨어 아키텍쳐

3 소프트웨어 아키텍쳐 개요

- 세부 내용보다는 중요한 부분만을 다룸
  - 중요한 부분은 주관적인 관점인데 이러한 주관적인 관점을 아키텍쳐에서 제공함

- 시스템 설계와 개발 시 적용되는 원칙과 지침이 있어야 함
  - 소프트웨어 아키텍쳐는 개발할 소프트웨어의 구조, 주요 구성 요소, 구성 요소의 속성, 구성 요소 간의 관계와 상호작용을 판단하고 결정함
  - 넓은 의미로는 사용자의 요구 사항을 충분히 반영한 소프트웨어의 목표와 프로그래밍 사이를 연결하는 것임

01 아키텍쳐와 소프트웨어 아키텍쳐

4 소프트웨어 아키텍쳐의 필요성

대규모 소프트웨어의 복잡성 해결의 첫 단계 역할

- 개발할 소프트웨어의 전체 구조를 가장 먼저 생각함
- 그 구조를 이루는 각 구성 요소를 찾음
- 각 구성 요소들 간의 명확한 관계를 설정함
- 일정한 규칙을 따름

01 아키텍쳐와 소프트웨어 아키텍쳐

5 잘 설계된 소프트웨어 아키텍쳐의 장점

- 사용자의 요구 사항과 완성된 프로그램 간의 연결을 최적화할 수 있음
- 최상위 수준의 아키텍쳐 설계에는 개발될 소프트웨어의 구조가 드러나지만, 구현과 관련된 세부 내용은 감추어져 있고 모든 기능적 요구 사항과 품질 요구 사항은 최대한 충족시킴
- 사용자가 요구하는 기능적인 요구 사항뿐 아니라 비기능적인 요구 사항도 만족시킴
- 모든 품질 요소 간에 절충을 제공함

2 소프트웨어 아키텍쳐의 특징과 기능

02 소프트웨어 아키텍쳐의 특징과 기능

1 아키텍쳐의 설계 시 고려 사항

① 의사소통 도구로 활용할 수 있어야 함

- 모든 이해 관계자에게 시스템의 공통된 추상화를 제공하여 이해를 돕고 의사소통 도구로 활용할 수 있게 함

② 구현에 대한 제약 사항을 정의해야 함

- 아키텍쳐는 개발 비용, 기간, 조직의 역량 등을 고려하여 조율함으로써 구현에 대한 제약 사항을 정의해야 함

02 소프트웨어 아키텍쳐의 특징과 기능

1 아키텍쳐의 설계 시 고려 사항

③ 품질 속성을 결정해야 함

- 모든 이해 관계자의 품질 요구 사항을 반영하여 시스템 품질 속성(성능성, 사용성, 보안성, 안전성, 검증성, 변경성 등)을 결정함

④ 재사용할 수 있게 설계해야 함

- 아키텍쳐는 특정 문제 영역에 적합한 소프트웨어 구성 요소를 표준화하고 패턴화하여 재사용할 수 있도록 설계해야 함

02 소프트웨어 아키텍쳐의 특징과 기능

2 아키텍쳐의 설계 시 기술 방법

① 이해하기 쉽게 작성

- 의사소통 도구로 활용되므로 사용자가 이해하기 쉽게 작성해야 함

② 명확하게 기술

- 애매모호한 표현을 사용하지 않고 명확하게 기술해야 함

02 소프트웨어 아키텍쳐의 특징과 기능

2 아키텍쳐의 설계 시 기술 방법

③ 표준화된 형식 사용

- 개발자가 명확하게 이해할 수 있게 표준화된 형식을 사용해야 함

④ 문서 버전 명시

- 문서의 버전을 정확히 명시하고 철저히 관리하여 혼란을 일으키지 않게 함

02 소프트웨어 아키텍쳐의 특징과 기능

3 아키텍쳐의 설계가 끝나면 할 수 있는 일

- 소프트웨어의 기본 골격이 만들어져 개발에 참여하는 사람들의 이해의 폭이 넓어지며, 구현상의 문제점을 도출할 수 있음
- 소프트웨어 아키텍쳐를 기반으로 분할 방법을 찾고 구조화를 위한 구체적인 방안을 생각할 수 있음
- 초기 설계의 원칙과 가이드를 제공할 수 있고, 상위 수준 설계를 재사용할 수 있음
- 소프트웨어 아키텍쳐를 기준으로 개발 조직을 만들 수 있음

02 소프트웨어 아키텍쳐의 특징과 기능

3 아키텍쳐의 설계가 끝나면 할 수 있는 일

- 전사 조직을 소프트웨어 아키텍쳐에 맞게 재편할 수 있음
- 품질 특성에 대한 평가 방법을 결정할 수 있음

3 소프트웨어 아키텍쳐의 품질 속성

03 소프트웨어 아키텍쳐의 품질 속성

1 소프트웨어 아키텍쳐의 품질 속성

- 품질 요구 사항
  - 시스템이 제공해야 하는 품질 속성의 수준
  - 가능하면 정확한 수치로 제시

- 소프트웨어 아키텍쳐
  - 이해 관계자들의 품질 요구 사항을 반영하여 품질 속성을 결정

03 소프트웨어 아키텍쳐의 품질 속성

1 소프트웨어 아키텍쳐의 품질 속성

- 품질 속성 기술 방법
  - 많은 품질 속성 중에서 해당 프로젝트에서 중요하게 생각하는 품질 속성을 결정
  - 정해진 품질 속성에 대해, 어느 정도 수준으로 설계할 것인지 목표를 설정
  - 그 목표를 달성할 수 있는 방법을 서술
  - 품질 속성 평가 방법을 서술

03 소프트웨어 아키텍쳐의 품질 속성

2 시스템 품질 속성

① 가용성(Availability)

- 시스템이 운용될 수 있는 확률로, 시스템이 장애 발생 없이 서비스를 제공할 수 있는 능력
- 가용성을 높이려면
  - 하드웨어 이중화처럼 여분의 구성 요소를 포함하도록 설계

03 소프트웨어 아키텍쳐의 품질 속성

2 시스템 품질 속성

② 변경 용이성(Modifiability)

- 변경 요구 사항을 받았을 때 쉽게 변경할 수 있는 능력
- 빈번하게 변경할 가능성이 높은 소프트웨어는
  변경 용이성을 고려하여 아키텍쳐를 결정

03 소프트웨어 아키텍쳐의 품질 속성

2 시스템 품질 속성

③ 성능(Performance)

- 사용자 요청과 같은 이벤트가 발생했을 때, 빠르고 적절하게 반응할 수 있는 능력
- 공유 자원을 어떻게 사용하는지, 어떤 알고리즘을 사용해 구현하는지 등의 요소와 밀접

03 소프트웨어 아키텍쳐의 품질 속성

2 시스템 품질 속성

④ 보안성(Security)

- 허용되지 않은 접근에 대응할 수 있는 능력

⑤ 사용성(Usability)

- 소프트웨어를 사용할 때 혼란스러워하거나 사용하는 순간에 고민하지 않게 하는 편의성

03 소프트웨어 아키텍쳐의 품질 속성

2 시스템 품질 속성

⑥ 테스트 용이성(Testability)

사용자가 요구하는 기능을 만족스럽게
잘 수행하고 있는지를 얼마나 쉽고 철저하게
테스트할 수 있는지를 나타냄

03 소프트웨어 아키텍쳐의 품질 속성

3 비지니스 품질 속성

① 시장 적시성(Time to market)

- 정해진 날짜에 소프트웨어를 출시해 경쟁력을 높일 수 있는 정도

  예) 서비스 개시 날짜를 이미 공표해놨거나 경쟁사의 제품 출시일을 미리 알고 있다면 경쟁력 우위를 위해 정해놓은 날짜에 맞추어 출시해야 함

03 소프트웨어 아키텍쳐의 품질 속성

3 비지니스 품질 속성

② 비용과 이익(Cost and benefit)

- 비용을 더 들여 사용하고 효과를 볼 것인지,
  아니면 비용을 절약하는 데 중심을 둘 것인지를 말함

- 아키텍쳐를 설계 시
  - 비용을 더 많이 들여 유연한 설계를 할 것인지,
    비용을 절감하는데 초점을 맞출 것인지 판단 필요

03 소프트웨어 아키텍쳐의 품질 속성

3 비지니스 품질 속성

③ 예상 시스템 수명(Predicted lifetime of the system)

- 수명이 중요한 경우라면 변경 용이성, 확장성, 이식성을 더 중요하게 고려

④ 목표 시장(Targeted market)

- 패키지 소프트웨어
  - 기능성 및 다양한 플랫폼에서도 잘 작동되어야 하므로 이식성을 충분히 고려한 설계 필요

03 소프트웨어 아키텍쳐의 품질 속성

3 비지니스 품질 속성

⑤ 신규 발매 일정 또는 공개 일정(Rollout schedule)

- 현재 버전에서는 기본 기능만 제공하고,
  추후에 배포할 차기 버전에서 기능을 추가하여
  완성도를 높일 예정이라면 유연성(Flexibility)과
  확장성을 고려한 설계 필요

⑥ 기존 시스템과의 통합(Integration with legacy system)

- 아키텍쳐 설계 시 기존 시스템과의 통합 방법을
  충분히 고려한 설계 필요

03 소프트웨어 아키텍쳐의 품질 속성

4 아키텍쳐 품질 속성

① 개념적 무결성(Conceptual integrity)

- 개념적 무결성은 일관성이라고도 함
- 전체 시스템과 시스템 구성 요소가 일관되도록 아키텍쳐를 결정

② 정확성과 완전성(Correctness and completeness)

- 사용자가 요구하는 기능을 충족시키는 정도로, 요구 분석 명세서와 일치하는 정도

03 소프트웨어 아키텍쳐의 품질 속성

4 아키텍쳐 품질 속성

③ 개발 용이성(구축 가능성, Buildability)

- 전체 시스템을 적절한 모듈로 분할한 후 개발 팀에 알맞게 분배하여 개발함으로써 정해진 기간 내에 완성하고, 개발 과정 중에도 쉽게 변경할 수 있는 능력

03 소프트웨어 아키텍쳐의 품질 속성

5 이해관계자별 품질 속성

① 발주자 관점

- 제품 가격(또는 개발비)이 중요, 응찰 시 가장 적게 써낸 업체 선정 확률 높음

② 사용자 관점

- 완벽한 기능뿐만 아니라 사용하기 쉽고 빨리 이해할 수 있는 아키텍쳐의 속성을 요구

03 소프트웨어 아키텍쳐의 품질 속성

5 이해관계자별 품질 속성

③ 개발자 관점

- 플랫폼이 달라져도 새로운 플랫폼에 쉽게 적용할 수 있는 아키텍쳐의 속성에 관심
- 변경 요청 시 쉽게 변경할 수 있는 설계