4장 서버 프로그램 구현

062 개발 환경 구축

- 응용 소프트웨어 개발을 위해 개발 프로젝트를 이해하고 소프트웨어 및 하드웨어 장비를 구축하는 것을 의미

하드웨어 환경 – 사용자와의 인터페이스 역할을 하는 클라이언트, 클라이언트와 통신하여 서비스를 제공하는 서버로 구성

클라이언트 종류 : 개인용 컴퓨터 / 스마트폰

서버의 종류 : 웹 서버 / 웹 애플리케이션 서버 / 데이터베이스 서버 / 파일 서버

소프트웨어 환경 – 클라이언트와 서버 운영을 위한 시스템 소프트웨어와 개발에 사용되는 개발 소프트웨어로 구성

시스템 소프트웨어 종류 : 운영체제 / 웹 서버 및 WAS 운용을 위한 서버 프로그램 / DBMS

개발 소프트웨어 종류 : 요구사항 관리 도구 / 설계/모델링 도구 / 구현 도구 / 빌드 도구 / 테스트 도구 / 형상 관리 도구

웹 서버의 기능 – HTTP/HTTPS 지원 / 통신 기록 / 정적 파일 관리 / 대역폭 제한 / 가상 호스팅 / 인증

개발 언어 선정 기준 – 적정성 / 효율성 / 이식성 / 친밀성 / 범용성

063 소프트웨어 아키텍처

- 소프트웨어를 구성하는 요소들 간의 관계를 표현하는 시스템의 구조 또는 구조체

모듈화 – 소프트웨어의 성능 향상, 시스템의 수정 및 재사용, 유지 관리 등이 용이하도록 시스템의 기능들을 모듈 단위로 나누는 것

추상화 – 전체적이고 포괄적인 개념을 설계한 후 차례로 세분화하여 구체화시켜 나가는 것

추상화의 유형 – 과정 추상화 / 데이터 추상화 / 제어 추상화

단계적 분해 – 문제를 상위의 중요 개념으로부터 하위의 개념으로 구체화시키는 분할 기법

정보 은닉 – 한 모듈 내부에 포함된 절차와 자료들의 정보가 감추어져 다른 모듈이 접근하거나 변경하지 못하도록 하는 기법

소프트웨어 아키텍처의 품질 속성 – 품질 평가 요소의 종류

시스템 측면 – 성능, 보안, 가용성, 기능성, 사용성, 변경 용이성, 확장성

비즈니스 측면 – 시장 적시성, 비용과 혜택, 예상 시스템 수명, 목표 시장, 공개 일정

아키텍처 측면 – 개념적 무결성, 정확성, 완결성, 구축 가능성, 변경성, 시험성

소프트웨어 아키텍처의 설계 과정

설계 목표 설정 -> 시스템 타입 결정 -> 아키텍처 패턴 적용 -> 서브시스템 구체화 -> 검토

064 아키텍처 패턴

- 아키텍처를 설계할 때 참조할 수 있는 전형적인 해결 방식 또는 예제

레이어패턴 – 시스템을 계층으로 구분하여 구성하는 고전적인 방법의 패턴

클라이언트-서버 패턴 – 하나의 서버 컴포넌트와 다수의 클라이언트 컴포넌트로 구성되는 패턴

파이프-필터 패턴 – 데이터 스트림 절차의 각 단계를 필터로 캡슐화하여 파이프를 통해 전송하는 패턴

모델-뷰-컨트롤러 패턴 – 서브시스템을 모델, 뷰, 컨트롤러로 구조화하는 패턴

기타 패턴 – 마스터-슬레이브 패턴 / 브로커 패턴 / 피어-투-피어 패턴 / 이벤트-버스 패턴 / 블랙보드 패턴 / 인터프리터 패턴

065 객체지향

- 소프트웨어의 각 요소들을 객체로 만든 후, 객체들을 조립해서 소프트웨어를 개발하는 기법

객체지향의 구성 요소 : 객체 / 클래스 / 메시지

객체지향의 특징 : 캡슐화 / 상속 / 다형성 / 연관성

객체 – 데이터와 이를 처리하기 위한 함수를 묶어 놓은 소프트웨어 모듈

클래스 – 공통된 속성과 연산을 갖는 객체의 집합

메시지 – 객체들 간의 상호작용을 하는데 사용되는 수단

캡슐화 – 외부에서의 접근을 제한하기 위해 인터페이스를 제외한 세부 내용을 은닉하는 것

상속 – 상위 클래스의 모든 속성과 연산을 하위 클래스가 물려받는 것

다형성 – 하나의 메시지에 대해 각각의 객체가 가지고 있는 고유한 방법으로 응답할 수 있는 능력

연관성 – 두 개 이상의 객체들이 상호 참조하는 관계

Is member of / is instance of / is part of / is a