소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 플랫폼 성능 특성 분석 기법
    - 사용자 인터뷰: 현행 플랫폼 사용자 인터뷰를 통해 속도의 적정성 확인
    - 성능 테스트: 현행 플랫폼을 대상으로 성능, 부하 테스트를 수행
    - 산출물 점검: 현재 플랫폼과 유사한 타사 제품의 성능 자료 등을 분석

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 플랫폼: 공급자와 수요자가 직접 참여해서 각자 얻고 싶은 가치를 거래할 수 있도록 만들어진 환경
- OS(운영체제): 사용자가 컴퓨터를 편리하고, 효과적으로 사용하는 환경을 제공하는 소프트웨어
- 네트워크: 컴퓨터들이 데이터 링크를 통해 서로의 데이터를 교환하는 기술
    - 네트워크 기본 모델(OSI 7계층) … 주어진 정의에 대해서 어떤 계층인지 알 수 있어야 함
        - 응용 계층: 사용자가 OSI 환경에 접근할 수 있도록 서비스 제공
        - 표현 계층: 코드 변환, 구문 검색, 암호화, 형식변환, 압축
        - 세션 계층: 전이중방식 또는 반이중 방식으로 종단 시스템의 응용 간 대화를 관리하는 계층
        - 전송 계층: 데이터 분할과 재조립, 흐름 제어, 오류 제어를 담당하는 계층
          : TCP와 UDP 프로토콜이 있다.
        - 네트워크 계층: 단말기 간 데이터 전송을 위한 최적화된 경로 제공
        - 데이터링크 계층: 물리적 연결을 이용해 동기화, 오류 제어, 흐름 제어 등의 전송 에러를 제어하는 계층
          : 물리 주소를 지정하고 흐름 제어 및 전송 제어를 수행하는 계층
        - 물리 계층: 매체 간의 인터페이스, 전기적, 기능적, 절차적 기능 정의

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 네트워크 현행 시스템 분석 활동
    - 플랫폼 기능/성능 특성 분석
    - 운영체제 분석
    - 네트워크 분석
    - DBMS 분석
    - 비즈니스 융합 분석

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 플랫폼: 공급자와 수요자가 직접 참여해서 각자 얻고 싶은 가치를 거래할 수 있도록 만들어진 환경
- OS(운영체제): 사용자가 컴퓨터를 편리하고, 효과적으로 사용하는 환경을 제공하는 소프트웨어
- 운영체제 분석 시 고려사항
    - 신뢰도
    - 성능
    - 기술 지원
    - 주변 기기
    - 구축 비용

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- DMBS: 데이터베이스 집합을 저장 및 관리하는 기능을 제공하는 프로그램
- DBMS 시스템 분석 시 고려사항
    - 가용성
    - 성능
    - 상호 호환성
    - 기술 지원
    - 구축 비용

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 비즈니스 융합: 정보통신기술을 적용해서 기존 제품을 혁신하기 위한 활동
- 비즈니스 융합의 유형
    - 고객 가치형: 개인 사회, 인류의 행복과 번영을 위한 가치 창출
    - 시장 유통형: 신시장 개척 또는 미래시장 선점
    - 가치 제안형: 시장/고객의 미 충족 욕구 대응 신상품 개발
    - 공급 역량형: 신기술, 신규역량을 활용한 상품생산 및 판매
    - 생산 방식형: 제품/서비스의 생산, 판매 프로세스 혁신

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 요구분석: 도출된 요구사항 간 상충을 해결하고, 외부환경과의 상호작용을 분석한다.
- 요구사항 분류
  - 요구사항이 기능인지 비기능인지 확인한다.
  - 우선순위가 동등하지 않으며, 중요도 / 시급도에 따른 서로 다른 우선순위를 부여한다.
  - 요구사항이 소프트웨어에 미치는 영향의 범위를 파악한다.
  - 요구사항이 제품에 관한 것인지 프로세스에 관한 것인지 확인한다.
- 요구사항 분석기법

  : 요구사항 분류 / 개념 모델링 / 요구사항 할당 / 요구사항 협상 / 정형 분석

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 요구분석: 도출된 요구사항 간 상충을 해결하고, 외부환경과의 상호작용을 분석한다.
- UML(Unified Modeling Language): 소프트웨어 개발과정에서 사용되는 모델링 언어
- UML의 특징: 가시화 언어 / 구축 언어 / 명세화 언어 / 문서화 언어
- UML의 구성요소
    - 사물
    - 관계
    - 다이어그램

        1) 요구사항: 유스케이스(Usecase) 다이어그램 (사용자 관점에서 시스템의 활동을 표현)

        2) 정적 모델링: 클래스 / 객체 / 컴포넌트 / 배포(Deployment) 다이어그램

            - 클래스: 시스템 내 클래스의 정적 구조를 표현

        3) 동적 모델링: 시퀀스 / 협업 / 활동 / 상태 다이어그램

            - 시퀀스: 객체 간 상호작용을 메시지 흐름으로 표현

            - 활동: 활동의 순서대로 흐름 표시

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 애자일(Agile) 방법론: 개발과 동시에 피드백을 받아서 유동적으로 개발하는 방법

  ⇔ 폭포수(Waterfall) 방법론: 처음부터 끝까지 계획을 수립하고 개발하는 방법

- 애자일 방법론 주요 원칙: 개인과 고객을 위하며, 계획 보다 변화에 대응한다.
- 애자일 방법론 유형
  - XP(eXtreme Programming)
    - XP의 5가지 가치: 용기 / 단순성 / 의사소통 / 피드백 / 존중
  - 스크럼
    - 스크럼 방법론 … 주어진 정의에 대하여 어떤 개념인지 알아야 한다.

      1) 제품과 프로젝트에 대한 요구사항: 백로그
      2) 2~4주의 짧은 개발 기간으로 반복적 수행을 통해 개발품질 향상: 스프린트
      3) 매일 15분 정도 미팅으로 To-Do List 계획을 수립: 데일리 미팅
      4) 프로젝트 리더, 스크럼 수행 시 문제를 해결하는 사람: 스크럼 마스터
  - 린
    - 린 개발방법론 7가지 원칙: 낭비제거 / 품질 내재화 / 지식 창출 / 늦은 확정 / 빠른 인도 / 사람 존중 / 전체 최적화

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 모델링: 실세계 현상을 쉽게 표현한 기법
- 모델링 절차: 요구사항 분석 > 개념 모델링 > 논리 모델링 > 물리 모델링
    - 개념 모델링의 종류: 유스케이스, 데이터 흐름, 상태 모델 목표기반, UI, 객체 모델, 데이터 모델

- 요구사항 관리 도구의 기능
    1) 기본 기능: 프로젝트 생성 / 요구사항 작성 / 요구사항 불러오기 및 내보내기
    2) 핵심 기능: 요구사항 이력 관리, 요구사항 베이스라인, 요구사항 추적성
    3) 부가 기능: 협업 환경 / 외부 인터페이스 / 확장성

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 분석 자동화 도구 : 요구사항을 자동으로 분석하는 도구(CASE: Computer Aided Software Engineering)

    1) 상위 CASE: 모델의 오류, 모순 검사 및 자료 흐름도 작성

    2) 중간 CASE: 상세 설계 작업, 화면 출력 작성 지원

    3) 하위 CASE: 시스템 명세서 및 코드 생성 지원

- 분석 자동화 도구에 대한 특징

    - 표준화 적용과 문서화를 통한 보고를 통해 품질 개선이 가능하다.
    - 변경사항과 변경으로 인한 영향에 대한 추적이 쉽다.
    - 명세에 대한 유지보수의 비용 축소가 가능하다.
    - 부넉 자동화 도구는 표준화를 통해 범용성과 이식성을 갖는다.

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- CASE에 대한 설명

  - 소프트웨어, 하드웨어, 데이터베이스, 테스트 등을 통합하여 소프트웨어를 개발하는 환경을 조성한다는 의미이다.
  - 프로그램의 요구 분석, 설계, 검사 및 디버깅 과정 전체 또는 일부를 자동화하는 것이다.
  - 소프트웨어 생명주기의 전체 단계를 연결해 주고 자동화해주는 통합된 도구를 제공한다.
  - 개발 과정의 속도를 향상시킨다.
  - 소프트웨어 부품의 재사용을 가능하게 한다.

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- UI(User Interface): 사용자와 시스템 사이의 의사소통이 가능하게 만드는 가상의 매개체
- UI 유형 … 주어진 정의에 대하여 어떤 인터페이스인지 알아야 한다.
    - CLI(command line interface) : 텍스트(명령어) 기반 인터페이스
    - GUI(graphical user) : 그래픽 환경 기반 인터페이스
    - NUI(natural user) : 직관적(사용자 가진 경험) 기반 인터페이스
    - OUI(organic user) : 유기적 상호작용 기반 인터페이스

- UI 설계 원칙 … 주어진 정의에 대하여 어떤 원칙인지 알아야 한다.
    - '용이한 검색, 쉬운 사용성, 일관성' 을 부특성으로 한다. : 직관성
    - 정확하고 완벽하게 사용자의 목표가 달성될 수 있어야 한다: 유효성
    - 초보와 숙련자 모두 쉽게 배울 수 있어야 한다: 학습성
    - 사용자의 인터랙션(의사소통과정)을 최대한 포용하고 실수를 방지할 수 있어야 한다: 유연성

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- UI 표준: 전체 시스템에 공통으로 적용되는 화면 규약
- UI 표준 적용을 위한 환경 분석
    - 사용자 트렌드 분석
    - 기능 조작성 분석
    - 오류 방지 분석
    - 최소한의 조작으로 업무 처리 가능여부 확인
    - UI의 정보 전달력 확인

- UI 표준 구성 요소
    - 전체적인 UX 원칙 / 정책 및 철학 / UI 스타일 가이드 / UI 패턴 모델 정의 / UI 표준 수립을 위한 조직 구성
        - UI 스타일 가이드 구성요소 : UI 구동 환경 / 레이아웃 및 네비게이션 / 기능 및 구성요소 정의
        - UI 패턴 모델 정의 : 서버 메시지 및 예외처리 정의

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- UI 스타일 가이드 구성

    1) 구동 환경 : 운영체제확인 / 웹 브라우저 확인 / 모니터 해상도 확인 / 프레임 세트 확인

    2) 레이아웃 정의 : 화면 구조 정의 / 상단, 좌측, 하단 메뉴 구성 / 내용 구성 / 사용 환경에 맞는 페이지 폭 정의

    3) 메뉴 내비게이션 타입 정의

    4) 기능 정의

    5) 구성요소 정의 : 그리드 / 버튼

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- UI 개발 기법 (UI 개발 목표 및 범위 정의 단계)
  - 3C 분석: customer, company, competition
  - SWOT 분석: strength, weakness, opportunity, threat
  - 시나리오 플래닝: 다양한 시나리오 설계를 통해 불확실성을 제거해 나감
  - 사용성 테스트: 사용자가 직접 제품을 사용하면서 질문에 답을 하는 테스트
  - 워크숍: 소집단이 서로 기술, 아이디어를 교환하며 검토하는 세미나

- UI 화면 설계 (위의 단계 완료 후 사용자 요구사항을 구체화하는 단계)
  1) 와이어 프레임: 간략한 흐름을 공유하기 위해 설계한 화면 단위의 레이아웃
  2) 스토리보드: UI 화면 설계를 위한 대부분의 정보가 수록된 문서
  3) 프로토타입: 스토리보드에 동적효과를 적용해서 실제 구현된 것처럼 시뮬레이션하는 모형

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- UI 설계 프로세스
    - 문제 정의 > 사용자 모델 정의 > 작업 분석 > 컴퓨터 오브젝트 및 기능 정의 > 사용자 인터페이스 정의 > 디자인 평가

- UI 흐름 설계
    - 화면에 표시 될 기능 작성 > 화면의 입력 요소 확인 > UI 요구사항 기반 유스케이스 설계 > 기능 및 양식 확인
        - (화면에 표시 될 기능)

        기능적 요구사항: 입출력 데이터 파악, 저장 데이터 분석, 수행 연산 분석

        비기능적 요구사항: 처리속도, 보안성과 같은 시스템 성능

- UI 상세 설계
    - UI 요구사항 기반 메뉴 구조 설계 > 화면과 폼 설계 > UI 검토 및 보완

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- UI 설계 도구: 가상의 매개체 UI의 설계를 지원하는 도구

1) 화면 설계 도구

- 파워포인트 / 파워 목업(실제 제품이 나오기 전에 만드는 모형) / 발사믹 목업 / 카카오 오븐

1) 프로토타이핑 도구

- UX핀, 액슈어(AXURE), 네이버 프로토나우(ProtoNOW)

2) UI 디자인 도구

- 스케치, Adobe XD

3) UI 디자인 산출물로 작업하는 프로토타이핑 도구

- 인비전, 픽사에이트, 프레이머

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 감성 공학: 인간의 감성을 평가하고 분석하여 제품으로 만드는 공학
    - 1류 접근 방법: 인간의 감성을 표현하는 어휘 이용
    - 2류 접근 방법: 개인이 갖고 있는 이미지를 구체화
    - 3류 접근 방법: 공학적 방법으로 수학적 모델을 구축하여 활용

- 프레이머: 커피 스크립트라고 하는 개발 언어를 사용하는 코드 기반의 프로토타이핑 도구, 코드 기반으로 작동되어 실제 작업물과 흡사하게 작동한다.

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 공통 모듈: 여러 프로그램에서 공통으로 사용할 수 있는 모듈(데이터와 함수의 묶음)을 의미한다.
- 공통 모듈의 원칙: 정확성 / 명확성 / 완전성 / 일관성 / 추적성
    - 정확성: 시스템 구현에 필요한지에 대해 정확하게 작성
    - 명확성: 해당 기능이 한 가지로 해석 될 수 있도록 작성
    - 완전성: 시스템 구현에 요구되는 모든 것을 기술
    - 일관성: 공통 기능 간에 상호 충돌이 없도록 작성
    - 추적성: 요구사항 출처와 시스템의 관계에 대한 식별이 가능하도록 작성

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 모듈화: 프로그램이 효율적으로 관리될 수 있도록 시스템 유지 관리를 쉽게 하는 기법이다.
- 모듈화 유형: 응집도, 결합도
    - 응집도: 모듈 내부 구성요소 간의 밀접한 정도 (응집도가 높을 수록 필요한 요소들로 구성되어 있는 것이다.)
    - 결합도: 모듈 간의 관련성 정도 (관련이 적을수록 모듈의 독립성이 높고, 모듈 간 영향이 적어진다.)

- 응집도 유형
    - 우연적 / 논리적 / 시간적 / 절차적 / 통신적 / 순차적 / 기능적 응집도

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 결합도 유형

  - 내용 / 공통 / 외부 / 제어 / 스탬프 / 자료 결합도

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 설계 모델링: 소프트웨어를 모델링하여 표현, 분석, 검증하는 과정
- 소프트웨어 설계 유형
    1) 자료 구조 설계: 소프트웨어를 구현하는데 필요한 자료 구조로 변환
    2) 아키텍처 설계: 소프트웨어 시스템 전체 구조 기술
    3) 인터페이스 설계: 상호 작용하는 컴퓨터 시스템, 사용자가 통신하는 법 기술
    4) 프로시저(procedure) 설계: 아키텍처 컴포넌트를 소프트웨어 컴포넌트 프로시저 서술로 변환

- 바람직한 소프트웨어 설계 지침
    - 적당한 모듈의 크기를 유지한다.
    - 모듈 간의 접속 관계를 분석하여 복잡도와 중복을 줄인다.
    - 효과적인 설계를 위해, 결합도(상관성)를 약하게 응집도를 강하게 함으로써 모듈의 독립성을 확보한다.
    - 모듈 간의 효과적인 제어를 위해 설계에서 계층적 자료 조직이 제시되어야 한다.
    - 효과적인 모듈 설계를 위해서 결합도를 약하게 해야 한다.
    - 모듈은 단일 입구와 단일 출구를 갖도록 설계한다.
    - 요구사항을 모두 구현해야 하고, 유지보수가 용이해야 한다.

소프트웨어 설계

요구사항 확인 > 화면 설계 > 애플리케이션 설계 > 인터페이스 설계

- 소프트웨어 아키텍처: 소프트웨어 구성요소 중에서 외부에 드러나는 특성, 시스템의 구조
- 소프트웨어 아키텍처 프레임워크: 아키텍처가 표현해야 하는 내용과 관계를 제공한다.
    - 구성요소: 아키텍처 명세서(기록) / 이해관계자(사람, 조직) / 관심사 / 관점 / 뷰(전체 시스템) / 근거

- 소프트웨어 아키텍처 4+1 뷰 개념: 고객의 요구사항 시나리오를 4개의 관점에서 바라보는 접근 방법이다.

    1) 유스케이스 뷰: 아키텍처를 도출하고 설계
    2) 논리 뷰: 시스템의 기능적 요구사항 지원
    3) 프로세스 뷰: 런타임 시, 시스템의 상호작용 관계를 표현
    4) 구현 뷰: 개발 환경 안에서 정적인 소프트웨어 모듈의 구성을 표현
    5) 배포 뷰: