SW 개발생명주기 및 형상관리

김예찬

1. 기존의 개발생명주기에 대한정리해보고, 최근 트랜드가 되고있는 개발방법론

1) 소프트웨어 개발 생명주기의 정의

- SDLC : Software Development Life Cycle
- 소프트웨어의 생성에서소멸까지 변환되는 과정
- 프로세스모델 또는 소프트웨어 공학 패러다임이라고도함
- 소프트웨어가 개발되기 위해 정의되고 사용이 완전히 끝나 폐기될때까지의 전과정을 단계별로나눈것으로, 조직내에서의 장기적인 개발계획과 개발과정중심의 관점

2) 개발방법론

- 폭포수모델

- 개념 정립에서 구현까지 하향식 접근 방법으로 추상화 Modeling 실행 하는모델

✓ 폭포수모델의 특징

- 가장 많이사용되었던 전통적인모델
- 소프트웨어개발을단계적, 순차적, 체계적 접근 방식으로수행
-각 단계별로 철저히 매듭짓고다음단계로 진행함
- 검토(Validation) 및검증(Verification), 검사(Test)에 의해 프로젝트 전반의 품질 향상 추구

- 프로토타입 모델

- 고객의요구를 불완전하게 이해하고 있거나 완벽한 요구분석의 어려움을 해결하기 위하여 실제 개발의 일부분만을우선개발(시험제작)하여 사용자와의 의사소통을 통해 타당성 평가 후 조정 및 진행이 되도록 하는모델

✓ 프로토타입 모델의 특징

- 사용자 요구를 더 정확히 반영 가능
- 시스템이해도가 낮은 관리자가 있는경우 유용함
- 개발중에도 유지보수효과가 있음

- 나선형 모델

- 소프트웨어의 기능을 나누어 점증적으로 개발하는 모델로 시스템을 개발하면서 생기는 위험을 최소화하기 위해 나선을 돌면서 점진적으로 완벽한 시스템으로 개발하는 모델

✓ 나선형 모델의 장점 및 단점
1) 장점
- 대규모 시스템 개발에 적합
-프로젝트의 완전성 및 위험감소와 유지보수의 용이
2) 단점
- 관리가 중요하나 매우 어렵고 개발시간이 장기화될 요지 있음
- 위험분석의 진행이 어려움
- 복잡한 프로세스에대한적응의어려움

✓ 나선형 모델의단계
- 계획수립: 목표, 기능선택, 제약 조건의 결정
- 위험분석: 기능선택의 우선순위 분석
- 개발: 선택된 기능의개발
- 평가: 개발 결과의 평가

- 증분 개발모델 (Incremental Development Model)
- 폭포수모델변형으로 하향식 구조의 수준별 증분의 분리개발, 이후 최종제품 통합
-증분 개발모델의 절차

- 진화적개발모델 (Evolutionary Development Model)
- 각 구성요소 핵심 부분의 개발, 개선, 발전시켜 나가는 발전적인 모델
- 진화적 개발모델의 개발절차

- RAD 모델

- 요구사항 정의, 분석 및 설계와 CASE를 사용하여 최소한의 활동에 의한 신속한 시스템 구축 개발 방법론

✓ RAD 모델의 특징
- 프로토타이핑방식 기준 사용자 요구사항, 분석, 설계, 개발을 신속한 시스템으로 개발
-제한된 범위의 단독시스템을 CASE와 같은 다양한 도구를 활용하여 신속히 개발함
- 고급언어의 모호성을 해결하기 위해 형식규격 언어로 표현하려는 노력이 진행
- 전통적인 SDLC의 사용자 참여미흡, 늦은 생명주기를 극복하기 위한 대안

2. 형상관리(버전관리)란 무엇이고, 형상관리(버전관리) 방법에는 어떤 것들이 있는가?

소프트웨어 형상 관리란 소프트웨어 소스 코드 뿐 아니라 개발 환경, 빌드 구조 등 전반적인 환경 전반적인 내역에 대한 관리 체계를 정의하고 있다. configuration이란 영어 원문에서의 뜻은 소프트웨어의 BOM(Bill of Materials)을 운용하는 체제를 얘기하고 있으며, 대부분의 산업 공학이나 생산 시스템에서 이야기하는 제조 공정을 소프트웨어 엔지니어링에 적용한 경우를 의미한다.
즉, 하나의 소프트웨어 산출물(binary)을 생성하기 위해 필요로 하는 아이템들과 공정 방식의 정의, 그리고 재생성을 위한 전반적인 환경까지 베이스라인(baseline)화하여 관리하는 방식 전체를 의미하며 이를 체계화한 사항을 형상 관리 시스템으로 정의하고 있다.

1) 버전 관리

- 버전관리는 소프트웨어 자산을 통제하고 변경의 위험을줄이며 파일과 프로젝트에 대한 History정보를 생성
- 버전의 생성, 재생성 및 삭제
- 버전 명에 기초한 오퍼레이션의 수행
- 버전 내용의 변경
- 개발 정보에 대한 리포트

2) 빌드관리

- 빌드관리는 스크립트의 작성 및 Build에 대한 관리이다.
- 프로그램 빌드에 대한 반복적인 절차를 정의 하는 능력
- 일관된 프로그램 빌드
- 모든 툴 형식의 제어

2) 변경요청 관리

- 변경요청 관리는 문제 및 변경요청의 추적이다.
- 개발에 대한 문제점과 변경요청 사항을 해결 과제로 연결
- 작업과 우선 순위를 할당하고 관리하는 능력
- 소프트웨어 릴리즈를 예고하고 노력을 단순화하며 가장 효과적인 진행과정과 사람들을 지적한다.

3) 릴리즈 관리

- 릴리즈 관리는 릴리즈를 정의하는 유연한 방법을 제시한다.
- 쉽게 사용자 릴리즈를 정의하는 능력

- 다양한 버그가 수정된 릴리즈로의 신속한 전환
- 분배를 위한 모든 성분들의 취합능력

4) 배포 관리

- 배포관리는 최종 제품의 배포 방법이다.
- 전자적인 소프트웨어 배포
- 인터넷
- 릴리즈 디스켓 혹은 CD-ROM

5) 설치 관리

- 설치관리는 적절한 설치와 소프트웨어의 환경을 지원한다.
- 적절하고 효율적으로 소프트웨어의 설치 및 환경 설정
- 시간 낭비없이 소프트웨어를 사용한다.