* 소프트웨어 생명 주기 : 개발을 위해 정의, 운용, 유지보수 등 과정을 단계별로 나눈 것
* 소프트웨어 공학(SE) : 품질&생산성 향상 목적, 현대기술적용, 지속적 검증, 명확한 기록
* 개발 방법론 세가지

|  |  |  |
| --- | --- | --- |
|  | 정의 | 특징 |
| 폭포수 모형 | 각 단계를 확실히 매듭짓고 결과 철저하게 검토하는 개발 방법론. | 가장 오래되고, 폭넓게 사용된 전통 모형, 생명주기 모형, 선형 순차적 모형 |
| 나선형 모형 | 점진적으로 완벽한 최종 소프트웨어 개발하는 것 | 폭포수 모형 + 프로토 타입 모형 장점 결합, 위험 분석 기능 추가, 위험 관리 최소화, 요구사항 첨가 가능, 정밀, 유지보수 필요없음. (계획 -> 위험 분석 -> 개발&검증 -> 고객평가) |
| 애자일 모형 | 고객의 요구사항에 유연하게 대응할 수 있도록 일정한 주기를 반복하면서 개발 | 고객과의 소통에 초점, 기업 활동 전반에 사용, 스크럼, XP, 칸반, 린, 크리스탈 등등. |

* 애자일 가치 : 개인 상호작용, SW(문서보다는 소프트웨어를), 고객, 협업, 변화에 반응
* 애자일 개발 모형

1. 스크럼 : 팀이 중심이되어 개발 효울성 높임. 모든 것을 스스로 해결, 제품책임자(PO) / 스크럼마스터(SM : 가이드 역할, 회의 주관, 공론화 처리) / 개발팀(DT: 7~8명)
2. XP : 고객 참여와 개발 과정의 반복을 극대화해 생산성 향상, 짧고 반복적인 주기, 단순한 설계, 빠르게 개발이 목적, \*5가지 핵심 가치(의사소통, 피드백, 존중, 용기, 단순성)

* 운영체제(OS) : 사용자가 컴퓨터를 효율적으로 사용할 수 있도록 환경을 제공하는 SW

( 고려사항 : 가용성, 성능, 기술 지원, 주변 기기, 구축 비용)

* DBMS : 데이터베이스 관리 시스템
* 웹 애플리케이션 서버(WAS) : 사용자의 요구에 따라 변하는 동적인 콘텐츠 처리 미들웨어
* 요구사항 정의 : 문제 해결을 위한 서비스 설명과 SW 정상 운영에 필요한 제약 조건
* 요구사항 유형

|  |  |
| --- | --- |
| 유형 | 내용 |
| 기능 요구사항  (시스템의 실질적 기능) | 시스템 수행 기능, 입력 출력, 데이터 저장, 연산, 사용자 요구 기능  (ex. 카카오톡 정산하기, 카카오톡 모임) |
| 비기능 요구사항  (시스템이 보다 만족스러울 수 있도록 만드는 기능) | 장비, 성능, 인터페이스, 보안, 품질, 제약사항, 프로젝트 관리, 지원  (ex. 정산 속도, 메모리 사용량 등등 |

* 요구사항 개발 과정 : 도(출) -> 분(석) -> 명(세) -> 확(인)

1. 도출: 식별 이해 과정 / 기법에는 인터뷰, 설문, 브레인스토밍, 워크샵, 유스케이스
2. 분석: 이해되지 않는 부분 걸러내기 위함 / 도구에는 자료흐름도(DFD), 자료사전(DD)

<요구사항 분석 도구>

|  |  |  |
| --- | --- | --- |
|  | 특징 | 기호 |
| 자료 흐름도(DFD) | 도형 중심, 흐름 그래프, 버블 차트 | 프로세스(시스템) - 사각형  자료흐름 – 화살표  자료 저장소 -  단말 – 교신 외부개체 |
| 자료 사전(DD) | DFD에서 더 자세히 정의하고 기록, 데이터를 설명하는 데이터 = 메타 데이터 | = : 자료 정의  + : and , ( ) : 생략  [ | ] : or, { } : 반복  \*\* : 주석 |

1. 명세: 문서화
2. 확인: 검토하는 활동

* 요구사항 명세 기법

|  |  |  |
| --- | --- | --- |
| 구분 | 정형 명세 | 비정형 명세 |
| 특징 | 수학적 원리 기반, 기호, 정형화, 간결함/일관성/완전성  사용자 이해도 낮음 | 상태, 기능, 객체중심, 자연어 기반, 서술, 다이어그램, 해석이 달라짐.  사용자와 의사소통 용이 |
| 종류 | VDM, Z, Petri-net, CSP | FSM, Decision Table, ER모델링, SADT |

* 요구사항 분석 CASE(자동화 도구)
* HIPO : 분석, 문서화 기법 (입력, 처리, 출력 기능을 나타냄)

(고유 모듈 분할하여 인터페이스를 계층구조로 표현 한 것을 히포차트라고 함)

* UML(언디파인모델링랭귀지) : 대표적인 객체지향 모델링 언어
* 관계 : 연관(Asso), 집합(Aggre), 포함(Compo), 일반화(General), 의존, 실체화
* 뷰 : 여러 관점에서 시스템을 가시화한 것.
* 정적 모델링(구조적 다이어그램) / 동적 모델링(행위 다이어그램)

<구조 다이어그램 = 정적>

|  |  |
| --- | --- |
| 클래스 | 클래스 사이의 관계 표현, 구조파악 문제점 도출 |
| 객체 | 인스턴스를 이용해 표현, 럼바우 기법에서 객체 모델링에 활용 |
| 컴포넌트 | 구현 모듈인 컴포넌트간의 관계, 구현단계에서 사용 |
| 배치 | 물리적 요소 위치 표현, 노드와 통신경로 표현, 구현단계에서 사용 |
| 복합체 | 내부 구조를 표현 |
| 패키지 | 모델 요소들을 그룹화한 패키지들의 관계 |

<행위 다이어그램 = 동적>

|  |  |
| --- | --- |
| 유스케이스 | 사용자 요구 분석, 사용자 사례로 구성 |
| 순차 | 상호작용하는 시스템이 주고받는 메시지 |
| 커뮤니케이션 | 순차에 더하여 객체들 간의 연관까지 표현 |
| 상태 | 럼바우 동적 모델링에 활용 |
| 활동 | 객체 처리 흐름을 순서에 따라 표현 |
| 상호작용 개요 | 제어 흐름 표현 |
| 타이밍 | 객체 상태 변화와 시간 제약을 명시적으로 표현 |

* 스테레오 타입<< >> : UML 기능에 추가적인 기능 표현 길러멧 이라는 겹화살표 사용

(포함, 확장, 인터페이스, 예외, 생성자)