소프트웨어공학 R1 서브노트

강북심화 1기

강북 심화반

이제이 PE, 정유나 PE

토픽 이름	소프트웨어 공학
분류	SW > 소프트웨어공학
키워드(암기)	소프트웨어 공학 지식 체계(SWEBOK)(Software Engineering Body of Knowledge), 소프트웨어 원리, 지식, 도구
암기법(해당경우)	비복변무순

기출문제

번호	문제	회차
1	1. 다음의 개념에 대하여 설명하시오. 가. 소프트웨어의 주요 특성과 공학의 발전원리 나. 소프트웨어 공학이 다루는 주제와 그 목표 다. 좋은 소프트웨어의 조건들과 개발에 영향을 미치는 요인들	123.관리.2.1
2	소프트웨어공학과 관련된 다음 문제에 답하라. 가. 소프트웨어 공학이란 무엇인가? 나. 소프트웨어 공학이 추구하는 5가지 목표는 무엇인가? 다. 이러한 목표를 달성하기 위하여 가장 중요한 것은 모듈화이다. 모듈화란 무엇이며 어떤 장점이 있는 가?	72.응용.2.5

I. 소프트웨어의 주요 특성과 공학의 발전원리

가. 소프트웨어 공학(software engineering)의 정의

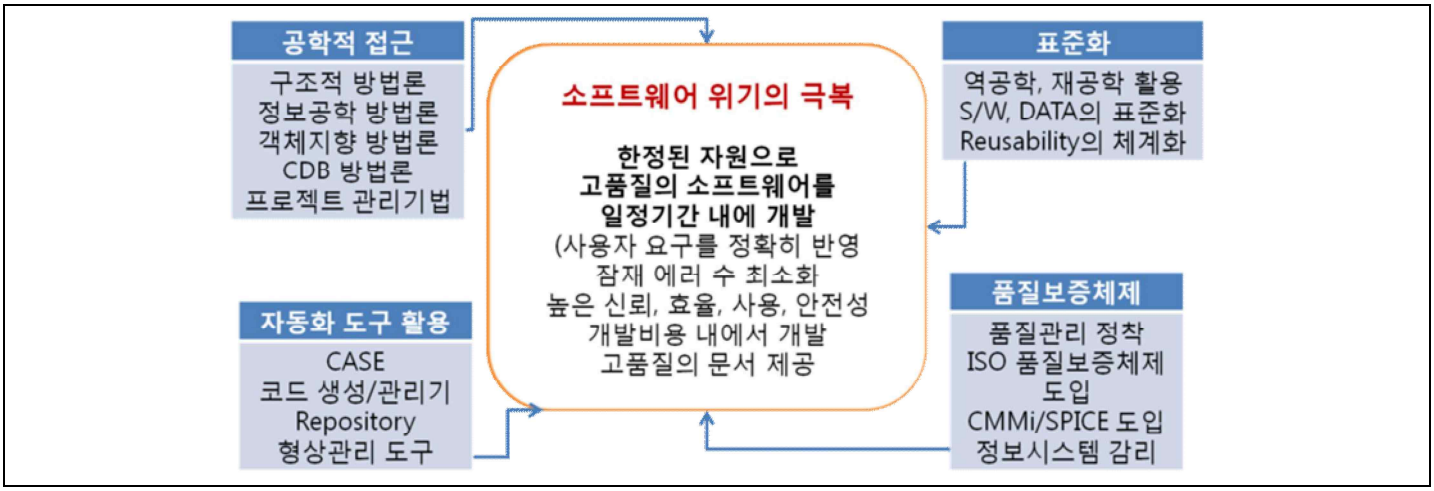
- 소프트웨어의 개발, 운용, 유지보수 등의 생명 주기 전반을 소프트웨어 원리, 지식, 도구 등을 적용하여, 체계적이고 서술적이며 정량적으로 다루는 학문으로, **소프트웨어 제품 개발에 공학적 기법을 적용한 공학**

나. 소프트웨어의 주요 특성(Brooks, 1871)

특성	설명
비가시성(Invisibility)	- 소프트웨어의 생산물 구조가 외부에 노출되지 않고 코드에 내재되어 있음
복잡성(Complexity)	- 정형적 구조가 없어 개발과정이 복잡하고 전산화 대상 업무, 소프트웨어 시스템 자체가 난해함. - 비규칙적, 비정규적
변경성(Changeability)	- 필요에 따라 항상 수정이 가능(진화성), 요구나 환경의 변화에 따라 적절히 변경
무형성(Intangible)	- 매우 중요하나 사실 형체가 없는 무형성 때문에 FP(Function Point) 등으로 유형화하려는 노력
순응성(Comformity)	- 사용자요구, 환경변화에 적절히 변형가능
복제 가능 (Duplicability)	- 소프트웨어는 간단하고 쉬운 방법으로 복제 가능, 다양한 경로와 노력으로 복제가 가능

- 시스템 대규모화에 따라 소프트웨어의 신뢰성 저하, 개발비의 증대, 계획의 지연 등의 현상이 현저히 발생하여 소프트웨어 위기 발생

다. 소프트웨어 공학(software engineering)의 발전 원리



- 소프트웨어 위기 극복을 위해 공학적 접근, 표준화, 자동화 도구, 품질보증체제 적용

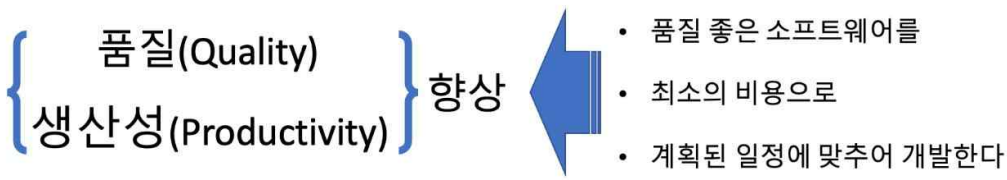
II. 소프트웨어 공학이 다루는 주제와 그 목표

가. 소프트웨어 공학이 다루는 주제

주제	의미	사례
방법(method)	- 소프트웨어 제작에 사용하는 기법이나 절차	- 구조적분석, 설계방법, 객체지향 분석, 설계 방법 - 설계도구
도구(tool)	- 자동화된 시스템	- 설계도구 - 프로그래밍 도구 - 테스트 도구
프로세스(process)	- 도구와 기법을 사용하여 작업하는 순서	- 애자일 - eXtreme Programming
패러다임(paradigm)	- 접근 방법, 스타일	- 구조적 방법론 - 객체지향 방법론

- 소프트웨어 공학의 적용을 통해 QCD(Quality, Cost, Delivery)의 만족을 목표로 함

나. 소프트웨어 공학의 목표

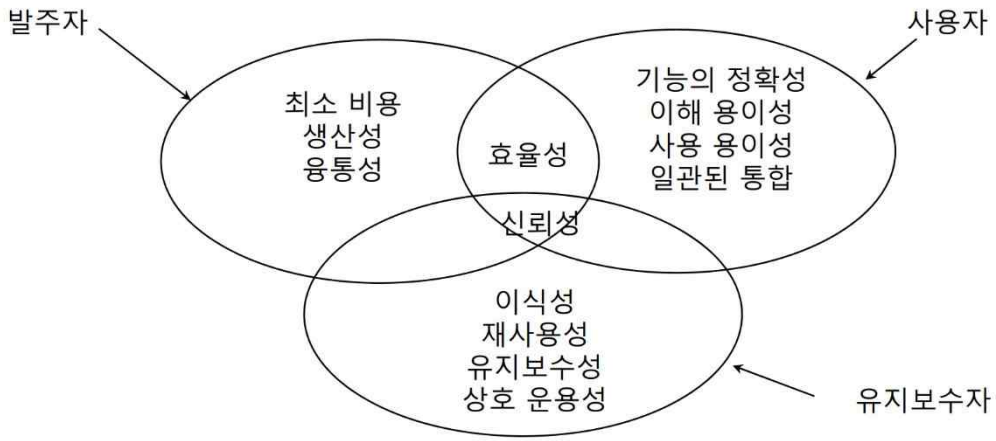


목표	필요 기법
고품질(Quality) 소프트웨어의 생산	- 요구사항 관리, 품질관리
사용자 만족도 증진	- 요구사항 관리, 품질관리
정해진 비용, 기간, 자원으로 소프트웨어 생산	- 정해진 비용, 기간, 자원으로 소프트웨어 생산
소프트웨어 생산 프로세스 수행능력 개선	- 요구사항 관리, 적정한 SDLC
생산성(Productivity) 향상	- 요구사항 관리, 부품화, 모듈화, 패턴화 기법

- 소프트웨어 공학의 목표는 좋은 품질과 생산성의 향상에 있음

III. 좋은 소프트웨어의 조건들과 개발에 영향을 미치는 요인들

가. 좋은 소프트웨어의 조건들



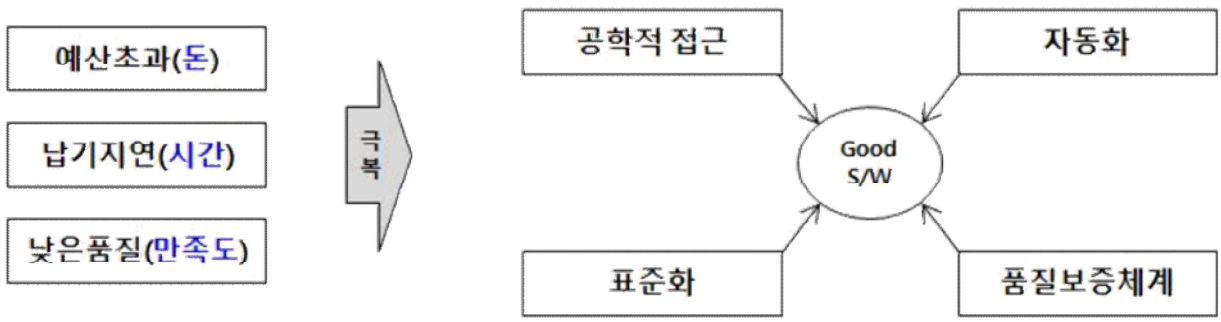
좋은 소프트웨어 조건	설명
정확성(Correctness)	- 기능적으로 맞게 동작하는가, 표준에 적합한가? - 요구 분석서의 기능과 일치하는지 점검
신뢰성(Reliability)	- 소프트웨어가 주어진 기간 동안 바르게 작동할 확률 - 오류 발생 확률에 반비례 - 정확성 제공하기 위한 필요조건
강인성(Robustness)	- 요구 명세에 표시하지 않은 상황(오류 입력)에서도 제대로 작동하는 성질
성능(Performance)	- 수행 속도, 데이터/트랜잭션 처리량 - 알고리즘의 시간 복잡도 - 시뮬레이션, 스트레스 테스트
사용 용이성(Usability)	- 시스템을 친근하게 느낄 수 있는 성질 - 사용 대상에 따라 달라질 수 있음 - 사용자 인터페이스, Human factor
유지보수성 (Maintainability)	- 보수성: 정해진 기간에 소프트웨어 결함을 해결할 수 있는 성질 - 진화성: 잠재적 발전 가능성 (추가 요구사항에 따라 기능이 진화할 수 있어야 함)
재사용성(Reusability)	- 소프트웨어 부품(라이브러리, 클래스 등)의 성질 - 확장 가능성(openness) - 적응성(adaptability) - 이용 용이성(closeness)

- 좋은 소프트웨어는 '사용자의 요구사항을 만족'하고 '정확하게 동작'하며 '쉬운 사용방법'과 '좋은 코드'로 개발된 소프트웨어임.

나. 좋은 소프트웨어 개발에 영향을 미치는 요인들

구분	영향 요인	설명
관리 측면	프로젝트 관리 기술	- 소프트웨어 개발 관리(프로그래밍 경험, 관리 능력) - 소프트웨어 프로세스 관리(일정, 예산, 인력, 형상, 품질 관리 등) - CMM(Capability Maturity Model) 모델 - 소프트웨어 품질 관리(Quality Assurance)
	프로젝트의 성격	- 응용분야에 따라 성격이 달라짐(자료처리 중심, 제어 중심, 시스템 소프트웨어, 인공지능) - 크기, 복잡도(소규모, 중규모, 대규모, 초대규모)
인력 측면	의사소통 (Communication skill)	- 발주자는 컴퓨터 및 소프트웨어에 대한 지식이 부족하고, 개발자는 발주자의 전문 분야에 대한 지식이 부족(인터뷰기술, 프로토타입) - 요구 취합 방법(설문지, 유저 그룹, 워크샵 등) - 정형적 방법
	프로그래머의 역량 (Maturity)	- (미숙한 프로그래머가 작성한 모듈은) 전체 품질이나 일정에 영향 - 프로그래머의 능력(프로그래밍, 커뮤니케이션, 응용분야에 대한 이해, 프로세스/도구에 대한 이해와 경험) - 소프트웨어 공학의 체계적이고 조직적인 접근법을 통하여 일정 부분 상쇄가 가능

IV. 좋은 소프트웨어 개발을 위한 제언



- 다양한 공학적 기법과 방법론, 관리체계를 기반으로 표준화를 통한 일관성과 호환성을 확보하고 자동화 툴을 사용하여 생산성 향상을 꾀하며 개발수명주기간 품질보증체계를 적용하여 개발초기부터 고품질 소프트웨어를 생산하도록 촉진하여 소프트웨어 위기라는 문제에 근본적 대응 필요

"끝"

토픽 이름	폭포수 모델
분류	SW관리 > SDLC > 폭포수모델
키워드(암기)	순차적, 산출물중심, 단계적 테스팅, 정식변경절차 수행(Frozen Delivery), 고전적 모델
암기법(해당경우)	

기출문제

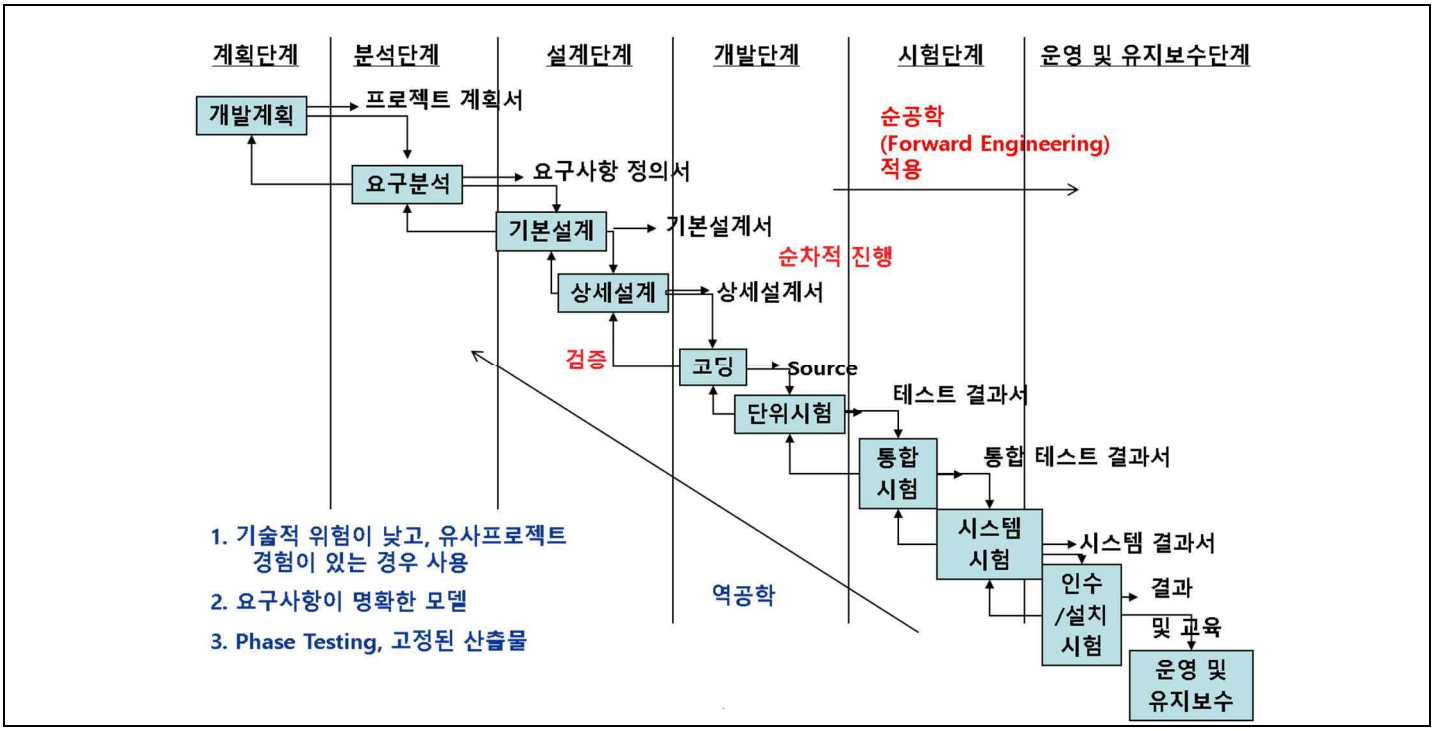
번호	문제	회차
1	폭포수 개발 방법론과 애자일 개발 방법론의 특징 및 장단점 비교	131.정보관리.3
2	반복점증적 개발방법과 폭포수형 개발방법을 비교하여 설명하시오.	105.정보관리.1

I. 고전적 선형/순차모델, 폭포수 모델의 개념

- 분석, 설계, 개발, 구현, 시험 및 유지보수 과정을 단계별로 구분하여 순차적으로 접근하는 방법
- 순차적, 하향식, 표준화된 양식 중심 프로젝트 관리 중시
- 산출물 중심 단계별 테스팅(Phase Testing), Frozen Delivery(정식 변경절차 수행) 강조

II. 폭포수모델 절차 및 장단점 분석

가. 폭포수 모델 절차



- SDLC 단계별 순공학 적용

나. 폭포수 모델 장단점 분석

구분	상세	내용
장점	관리 용이	- 간결하고 이해하기 쉬움
	체계적 문서화	- 단계별 정형화된 접근법으로 체계적 문서화 가능
	변화가 적은 프로젝트	- 요구사항 변화가 적은 프로젝트에 적합

		- 비교적 소규모 프로젝트 개발에 유리
단점	단계적 진행	- 앞 단계가 끝날 때까지 대기, 개발완료전에 사용자가 원하는 것을 정확히 알 수 없음
	오류/변경에 취약	- 단계 결과물이 완벽하지 않으면 다음 단계에 오류가 전파, 요구사항 변경시 전체일정에 부담

Ⅲ. 폭포수 모델의 단점 극복을 위한 프로토타이핑 모델과 비교

구분	폭포수 모델	프로토타이핑 모델
특징	- 앞 단계 종료 후 다음단계 진행	- 시제품 승인 후 본 제품 개발
	- 단계 별 정의 및 산출물이 명확	- 폭포수 모델 단점 극복 위해 제시
효과적 적용 유형	- 기술 위험이 낮고, 유사한 프로젝트 경험이 있는 경우	- 유사 프로젝트 경험이 없고, 기술 위험이 높은 경우

Ⅳ. 폭포수 모델과 애자일 모델의 비교

구분	폭포수 모델(Waterfall)	애자일 모델(Agile)
핵심요소	- Phase 단계별 수행, 계획 중심	- Less Document-Oriented, 학습 중심
요구사항	- 미리 정의된 요구 사항을 수집, 분석, 디자인	- 프로젝트 과정에 걸쳐 진화하는 요구사항 반영
Release	- 빅뱅(Big Band) Release - 요구사항을 제대로 따르면 기능적으로 잘 작동할 것으로 예상하여 통합을 자주하거나 빠르게 하지 않음	- 빠른 Release - 작은 크기로 Release된 기능들을 목표로 하고 있으며 고객 Needs를 파악하여 가능한 빨리 고객에게 제공하고 고객 요구 사항을 충족
의사소통	- 고객과 드문 의사 소통 - 문서 중심의 접근법으로 이해관계자들 사이 계약서로 문서 제공하여 의사 소통 최소화	- 고객과의 지속적인 의사 소통 - 작업본을 주기적으로 발전시키고 고객과 지속적으로 공유. 고객과 개발팀은 공통으로 업무를 진행하며 협업
개발구조	- 수평적 단계별 개발 - 기초에서 시작하여 단계를 거쳐 튼튼한 기반에 서비스 개발	- 기능별 수직 개발 - 서비스에 언제든지 요구사항을 추가 가능하므로 가능한 빠르게 개발하고 Feedback을 받아 그것을 반영하는 과정 반복
통합	- 마지막에 통합 - 계획에 맞춰 모든 것이 완벽이 작동하는 마지막 단계에서 통합	- 잦은 통합 - 통합 이슈와 같은 문제들을 빠르게 인지할 수 있도록 자주 통합 수행
테스트	- 마지막 단계 테스트 실시 - 마지막 전까지 통합하지 않으므로 마지막까지 테스트 수행 불가	- 초기와 이후 잦은 테스트 - 초기 통합하는 이유는 잠재하는 결함과 예상 못한 사용 패턴을 찾을 수 있어 수정에 대한 비용 절감

"끝"

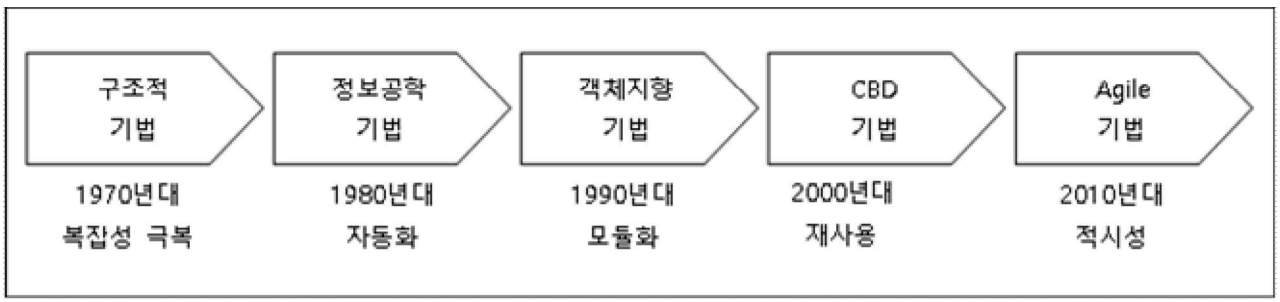
토픽 이름(상)	SW개발방법론
분류	SW > 개발방법론> SW개발방법론
키워드(암기)	방법/절차/도구 등을 체계화/표준화, 구조적, 정보공학 객체지향, CBD, Agile
암기법(해당경우)	구정객CA(구조, 정보, 객체, CBD, Agile)

기출문제

번호	문제	회차
1	4. 안전한 소프트웨어 개발 관련하여 다음에 대하여 답하시오. 가. 보안 소프트웨어 개발 생명주기(Secure SDLC) 방법론의 유형을 설명하시오. 나. 다음 조건의 Java 코드에서 보안 위험성이 없는 안전한 코드로 수정하시오. 조건) String param = request.getParameter("id"); ... String sql = "select name from board where id ='" + param + "'"; Connection con = db.getConnection(); Statement stmt = con.createStatement(); ResultSet rs = stmt.executeQuery(sql);	121.관.4
2	Agile 프로세스가 국내에 많이 도입이 되고 있다. 프로젝트 관리자는 개발방법론과 Agile 프로세스들 프로젝트에 맞게 테일러링(Tailoring)하고자 한다. EVM(Earned Value Management)와 Burn Down Chart를 비교하고, 프로젝트 적용 방안을 설명하시오	119.정.2.5
3	1. 소프트웨어 개발방법론의 4가지 유형을 설명하고, 그 중 하나인 제품계열(Product Line) 방법론의 필요성과 2가지 구성요소에 대하여 설명하시오. (118정4)	118.정.4.1

I. 소프트웨어의 품질과 생산성 향상을 위한 기반, SW 개발방법론의 개요

가. 개발방법론(SW Development Methodology)의 정의



- 소프트웨어를 생산하는 데 필요한 **프로그래밍 개발 과정**들을 정리하고 **표준화**하여 프로그래머들이 프로그래밍 개발과정에서 각 개인이 개발과정에서의 일관성을 유지하고 프로그래머들 간의 효과적인 협업이 이루어질 수 있도록 돕기 위한 **방법론**
- 소프트웨어 개발에 관한 계획, 분석, 설계 및 구축에 관련 정형화된 작업활동, 절차, 산출물, 기법을 방법을 공학적 기법으로 체계적으로 정리하여 표준화한 이론

나. SW 개발 방법론의 목적

구분	설명
작업표준화	개발 경험축적 및 재활용을 통한 개발생산성 향상
수행공정 가시화	가시적인 공정을 통해 효과적인 프로젝트 관리
의사소통수단	정형화된 절차와 표준용어 제공으로 의사소통 원활

II. 개발방법론의 주요 구성요소

구성요소	설명	예시
절차	- 프로젝트 단계별 활동 및 순서	Phase-Activity-Task
방법	- 각 Task의 수행방법(누가 무엇을 어떻게)	작업 방법
산출물	- 산출물의 목록 및 양식	설계서 등
관리	- 계획, 일정, 품질 등의 관리 방법	계획서, 기준문서 등
기법	- 각 단계별 이용가능한 기술 및 기법	ERD, DFD 등
도구	- 각 단계 또는 기법에 활용가능한 도구	CASE, UML Tool 등

III. 개발방법론 비교

구분	구조적방법론	정보공학방법론	객체지향방법론	CBD방법론
개념	- 정형화된 분석절차 적용. - **프로세스** 중심	- CASE 도구 등 공학적 접근 - **데이터모델** 중심	- **객체지향** 개념 적용, 사용자관점 분석설계	- **컴포넌트** 개발 및 조합을 통한 재사용 중심
시기	- 1970 년대	- 1980 년대	- 1990 년대	- 2000 년대
중점	- 기능중심 - 정형화된 분석 절차 적용 - 프로세스 중심	- 자료구조 중심 - 데이터모델 중심 - CASE 도구 등 공학적 접근	- 객체 중심 - 사용자 관점의 분석설계	- 컴포넌트 중심
특징	- **분할과 정복** - 하향식 기능분해	- 데이터와 프로세스 균형 - 기업정보시스템중심	- (**캡추다정상**) - 분석초점이 명확 - White Box Reuse	- 반복, 점진적 - 높은 재사용성 - 생산성/품질향상, 유지보수 최소 - Black Box Reuse
장점	- 프로세스 중심 방식 개발 유용	- 자료중심으로 비교적 안정적	- 자연스럽고 유연함 - 재사용성 향상	- 생산성, 품질, - 비용위험 개선 - SW 위기극복
단점	- 기능은 불안정한 요소 - 데이터 정보은닉 불가 - 낮은 재사용, 유지보수성	- 어플리케이션은 여전히 기능적 설계 - 기능의 유지보수, 재사용성 낮음	- 실질적인 재사용성 어려움 - 기본적 SW기술 필요	- 컴포넌트 유통 환경 개선필요 - 테스트 환경의 부족 - 컴포넌트평가, 인증 환경 미흡
목표	- 비즈니스 프로세스자동화	- 경영전략적 정보시스템 구축	- 재사용 시스템	- 컴포넌트 개발 및 활용
산업 구조	- 소품종 다량생산	- 다품종 소량생산	- 인터넷 비즈니스	- 인터넷 비즈니스
접근 방법	- 프로세스 중심	- 데이터 중심	- 객체중심 - (프로세스+데이터)	- 컴포넌트 중심
Life cycle	- 폭포수 모델	- 폭포수 모델	- 반복적 개발	- 반복적 개발

			- 프로토타이핑 - 기업정보시스템 중심 - AP가 데이터구조에 종속		
모델링		- 기능모델링	- 데이터모델링, - 프로세스모델링	- 객체모델링	- 객체모델링, 컴포넌트 모델링
개발 방식		- Top-down	- Top-down	- Bottom-up	- Bottom-up
자동화		- 수작업가능	- 자동화도구 요구	- 자동화도구 필요	- 자동화도구 필수
단계별 산출물	**계획**	- 도메인분석, 프로젝트 계획서	- 도메인분석, - 프로젝트 - 계획서	- Biz Process / Concept Model, 프로젝트 계획서	- Biz Process / Concept Model, 프로젝트 계획서
	분석	- Data Flow Diagram	- ERD, 기능차트, - Event 모델	- Use Case DM, Sequence DM, Class DM - DM=Diagram	- Use Case DM, Business type DM, Component DM, 재사용 계획서
	설계	- Structure chart, 프로그램 사양서	- 어플리케이션 - 구조도, - 프로그램사양서 - Table정의서/목록	- Sequence DM, Class DM, Component DM, Deployment DM	- Sequence DM, Class DM, Component DM
지원 언어		- Cobol, C, VB, Pascal	- Cobol, C, VB - Pascal	- C++, Java, C#, VB	- 원칙적으로 개발언어 무관