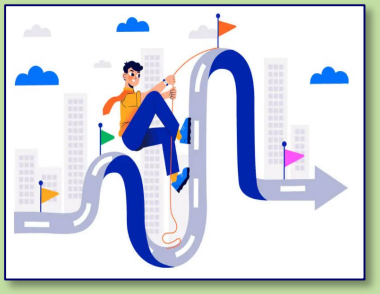


소프트웨어 프로세스 및 그 모델들

가을, 2024

제홍@충북.ac.kr

다루는 주제

소프트웨어 프로세스 모델

-폭포
-진화적(프로토타입, 증분적)
-애자일 모델(XP, 스크럼)
-변환
-나선형 모델
-V 모델
-DevOps 모델, ...

프로젝트 특성별 SDLC 소프트웨어 프로세스 표준 : ISO 12207

여행가자

최종 목적지까지 가는 방법은?

거기에 무엇이 필요한가요?

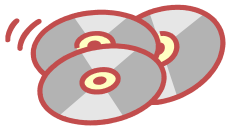
소프트웨어 개발은 어때요?

소프트웨어 개발 프로세스

소프트웨어 개발은 지속적인 절차적 활동입니다. 그 활동은 무엇입니까?

요구 사항

최종 시스템



소프트웨어 개발을 위한 다양한 경로를 어떻게 표현할 수 있을까?

- _____

소프트웨어 프로세스 모델이란?

소프트웨어 생산 프로세스

-소프트웨어 제품을 빌드, 전달, 배포 및 발전시키기 위해 따르는 프로세스

-아이디어의 시작부터 시스템의 전달 및 최종 폐기까지

생산과정의 목표

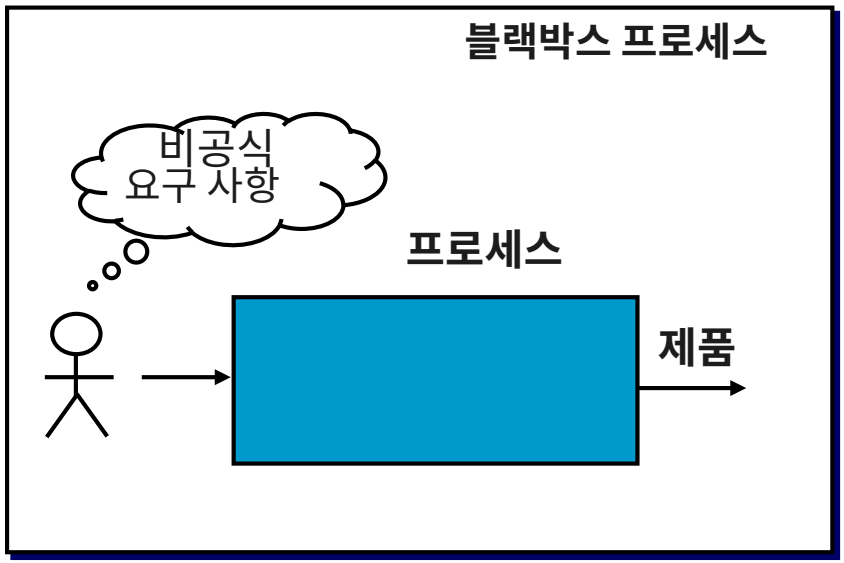
-고객의 기대에 부응하다

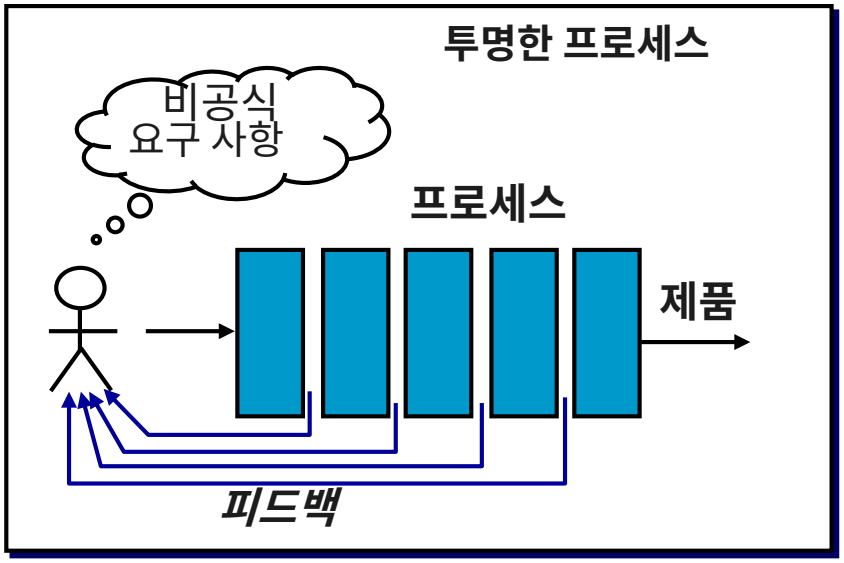
- 시간과 예산 내에서 고품질 제품을 제공함으로써
- 제품을 수익성 있게 만들어서,
- 생산을 안정적이고 예측 가능하며 효율적으로 만들어서

소프트웨어 수명 주기 모델

-요구 사항→분석→설계→코딩→ (테스트)→배달→ 운영 및 유지 관리→퇴직

프로세스 모델이 중요한 이유는?





출시 시간 단축 및 생산 비용 절감 프로세스는 제품 품질에 결정적인 영향을 미칩니다.

프로세스를 제어함으로써 필요한 제어를 더 잘 달성할 수 있습니다.

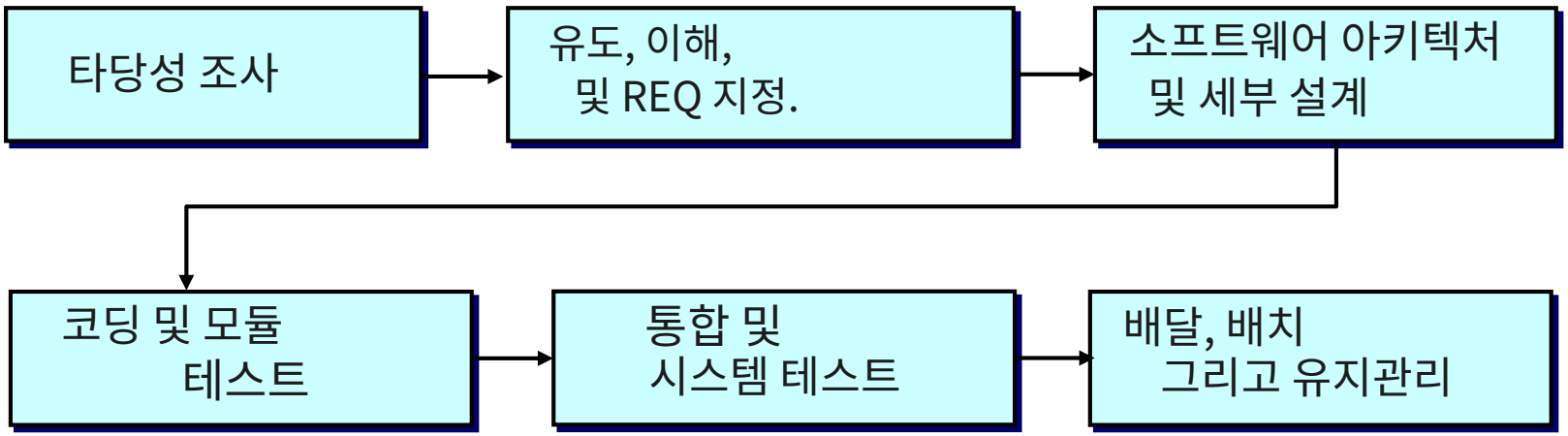
제품의 품질.

소프트웨어 생산 활동

프로세스

-무엇을 해야 하는지에 대한 일련의 (관련된) 단계

주요 활동



소프트웨어 프로세스 모델

소프트웨어 프로세스 모델

-소프트웨어 개발 활동이 프로세스로 어떻게 구성될 수 있는지에 대한 표현

대표적 프로세스 모델

-빌드 및 수정(코드 및 수정) 모델

-폭포수 모델

-진화 모델

- 신속한 프로토타입 제작
- 증분형

-변환 모델

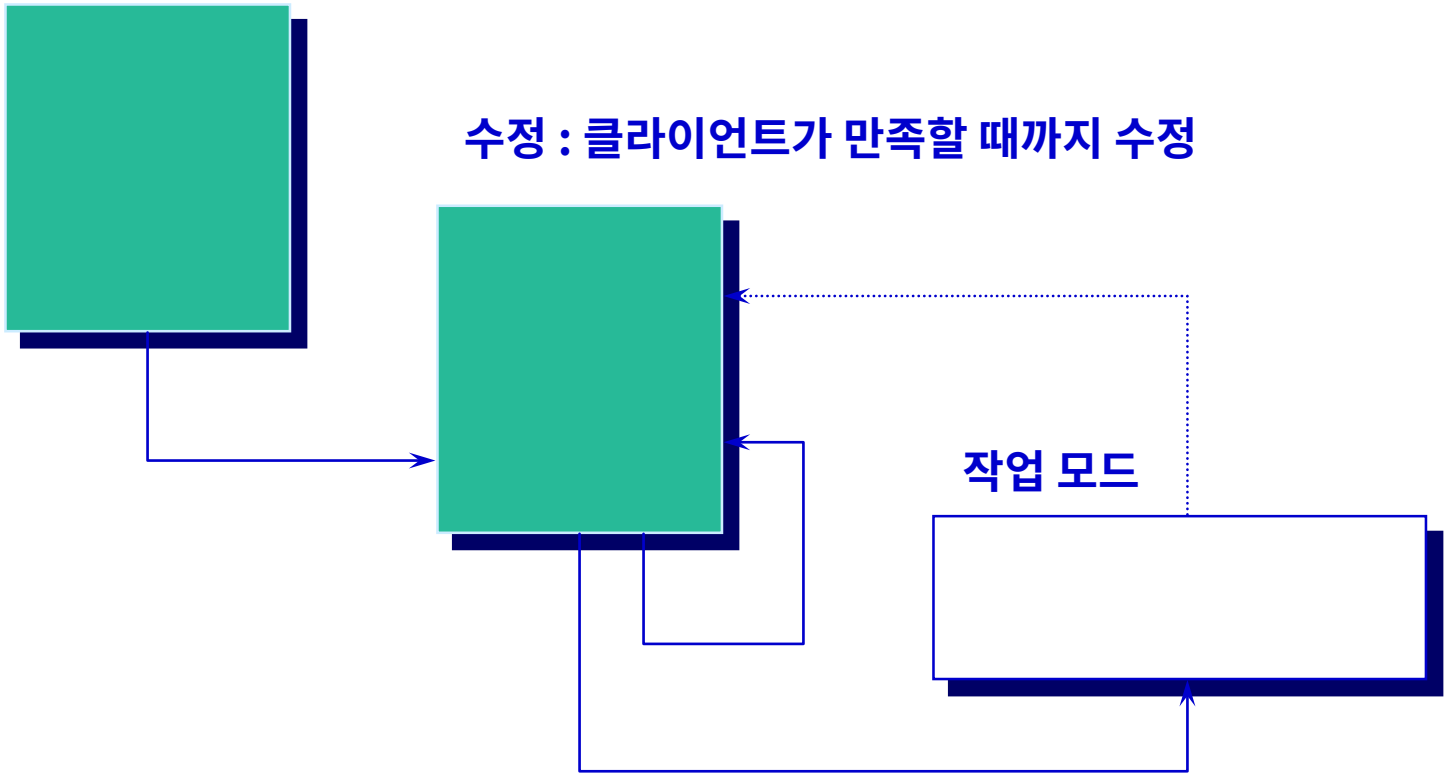
-나선형 모델

-기타

빌드 앤 픽스 모델

첫 번째 버전 빌드 : 코드 작성

수정 : 클라이언트가 만족할 때까지 수정

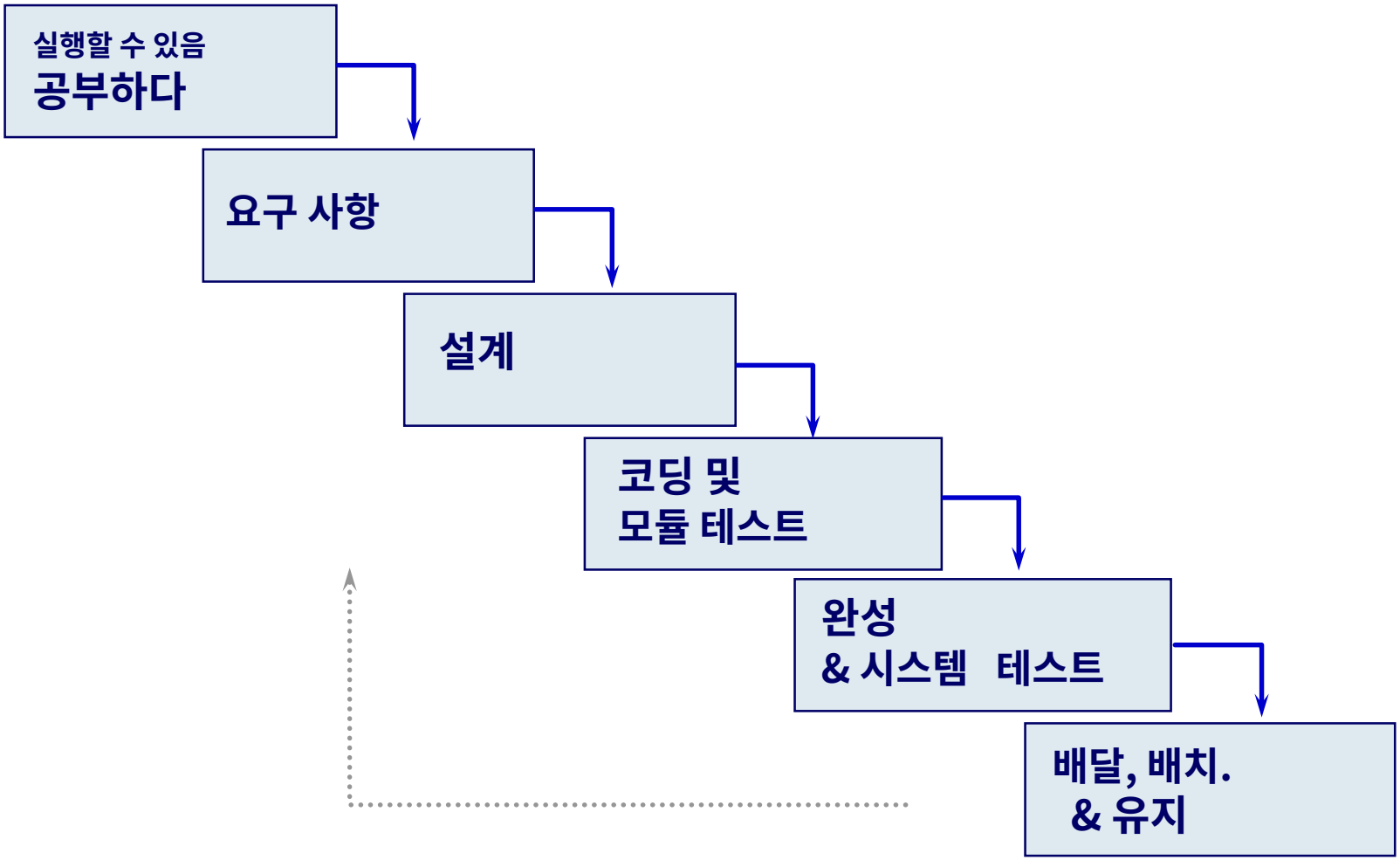


모델 구축 및 수정(계속)

주로 한 사람이 하는 작업입니다. 사양 없이 제작되었습니다. 적당한 크기의 제품에는 만족스럽지 않습니다. 오늘날의 환경에는 적합하지 않습니다.

-컴퓨터 지식이 없는 사람들을 위해 개발되었습니다.
-더욱 엄격한 신뢰성 요구 사항
-그룹 활동

폭포수 모델



폭포수 모델(계속)

1970년대부터 대중화된 산업 관행. 순차적, 단계 기반, 문서 지향 한 단계의 출력은 다음 단계의 입력을 구성합니다.
기여

-엄격하고 계획적이며 관리 가능한 접근 방식을 적용합니다.

-제품 구현은 해당 목적을 충분히 이해한 후로 연기하는 것이 좋습니다.

문제점은 무엇인가? 많은
변형이 존재합니다.

-피드백 루프를 갖춘 폭포수 프로세스

-증분형 빌드를 사용한 폭포수 프로세스

-등등…

진화 모델

단계가 증분 확장으로 구성된 모델
운영 소프트웨어 제품

개발 전략

-(실제 사용자에게 무언가를) 전달하다

-측정(모든 중요한 차원에서 사용자에게 추가된 가치)

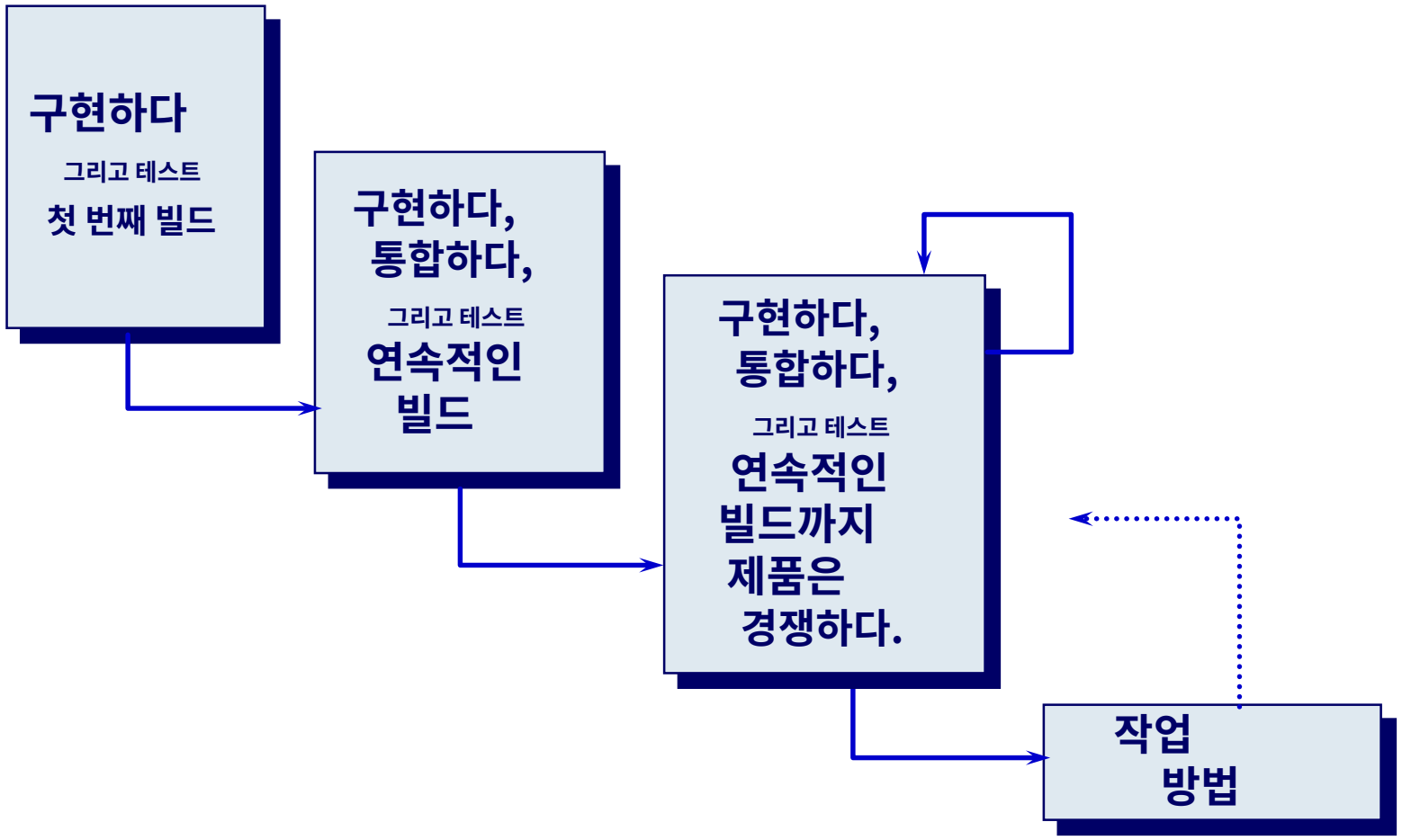
-(관찰된 현실에 기반한 디자인과 목표 모두) 조정

유지관리는 수명주기의 한 단계로 사라집니다. 진화모델의 두 가지 유형

-증분적 접근 방식

-프로토타입 제작

점진적 접근 방식


구현하다
그리고 테스트
첫 번째 빌드
구현하다,
통합하다,
그리고 테스트
연속적인
빌드
구현하다,
통합하다,
그리고 테스트
연속적인
빌드까지
제품은
경쟁하다.
작업
방법

증분적 접근 방식(계속)

단계적 개발

폭포수 모델에서 도입된 규율을 유지해야 합니다.
각 단계: 미니 폭포수 모델 프로세스의 시퀀스

이익

-사용자에게 새로운 제품에 적응할 수 있는 시간을 제공합니다.

-변화에 쉽게 적응 가능

-단계적 배송은 대규모 자본이 필요하지 않습니다.

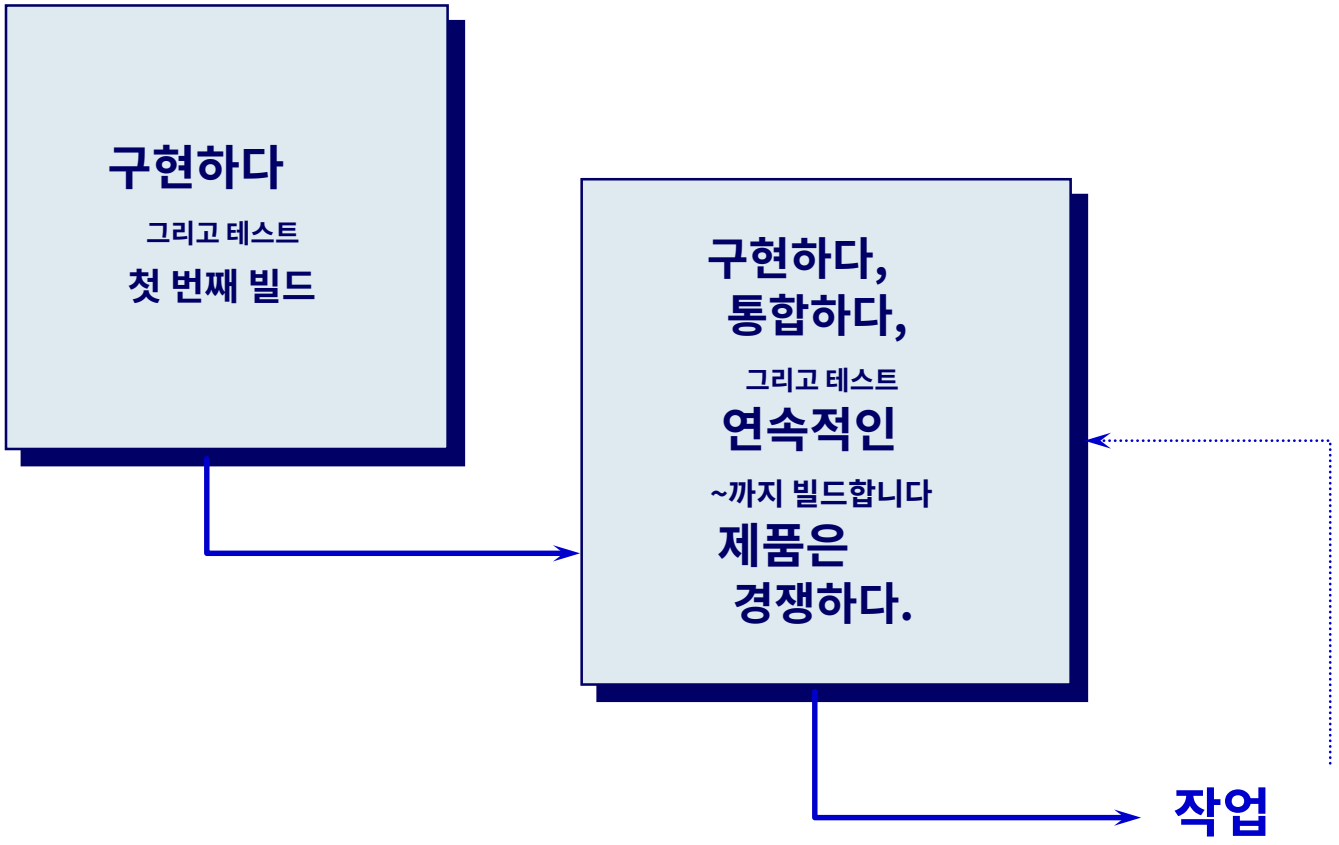
문제들

-빌드 및 수정 모델과 유사할 수 있음

-각 통합 및 테스트의 오버헤드

-사용자는 부분 시스템을 최종 시스템으로 간주할 수 있습니다.

(빠른) 프로토타입 제작



프로토타입 제작(계속)

두 번 해보세요

-첫 번째 버전

- 제품의 실현 가능성을 평가하기 위한 일회용 프로토타입
- 요구사항을 확인하기 위해

-두 번째 버전

- 폭포수 모델을 따르다

프로토타입은 점차 최종 시스템으로 진화할 수 있습니다.

이점

-비용 및 시간 절감에 도움이 됩니다

-개발자 간, 개발자와 사용자 간의 커뮤니케이션을 개선합니다.

-오류를 조기에 감지하는 데 도움이 됩니다.

문제

-변화를 예상할 필요성을 강조하지 않습니다.

신속한 애플리케이션 개발

Q1 : 어떤 방법을 사용할 수 있나요?빠르게 발전하다소프트웨어 시스템?

질문 2 :왜우리는 소프트웨어를 빨리 개발할 수 있나요?

신속한 애플리케이션 개발

빠른 개발 및 제공지금은 종종 가장 중요한 소프트웨어 시스템에 대한 요구 사항

- 기업은 빠르게 변화하는 요구 사항에 따라 운영되며 안정적인 소프트웨어 요구 사항 세트를 생성하는 것은 사실상 불가능합니다.
- 소프트웨어는 변화하는 비즈니스 요구를 반영하기 위해 빠르게 발전해야 합니다.

급속한 소프트웨어 개발 속에서

- 사양, 설계 및 구현은 서로 연결되어 있습니다.
- 시스템은 버전 평가에 이해관계자가 참여하는 일련의 버전으로 개발됩니다.
- 사용자 인터페이스는 종종 IDE와 그래픽 도구 세트를 사용하여 개발됩니다.

빠른 개발 - 애자일 개발

애자일 방법

대표적인 Agile 방법같은 문제를 목표로 함:

신뢰할 수 있는 소프트웨어를 더욱 빠르게 만들 수 있습니다.

- **동적 시스템 개발 방법**(데인 포크너 등)
- **적응형 소프트웨어 개발**(짐 하이스미스)
- **크리스탈 클리어**(방법의 가족, Alistair Cockburn)
- **XP(켄트 베크, 에릭 감마 등)**
- **스크럼(켄 슈와버, 제프 서덜랜드, 마크 비들)**
- **기대다**소프트웨어 개발(Mary와 Tom Poppendieck)
- **기능 중심 개발**(피터 코드와 제프 델루카)
- **Agile 통합 프로세스**(스콧 앰블러)

XP 프로세스

1990년대 Kent Beck의 eXtreme Programming 접근

방식 요구 사항 변경 처리

역할 및 책임

-프로그래머: 분석, 설계, 테스트, 코딩 및 통합

-관리자: 프로젝트 프로세스 진행 상황을 제어합니다.

-고객: 요구 사항 및 우선순위

4가지 가치를 추구하다

-의사소통

-간단

-피드백

-용기

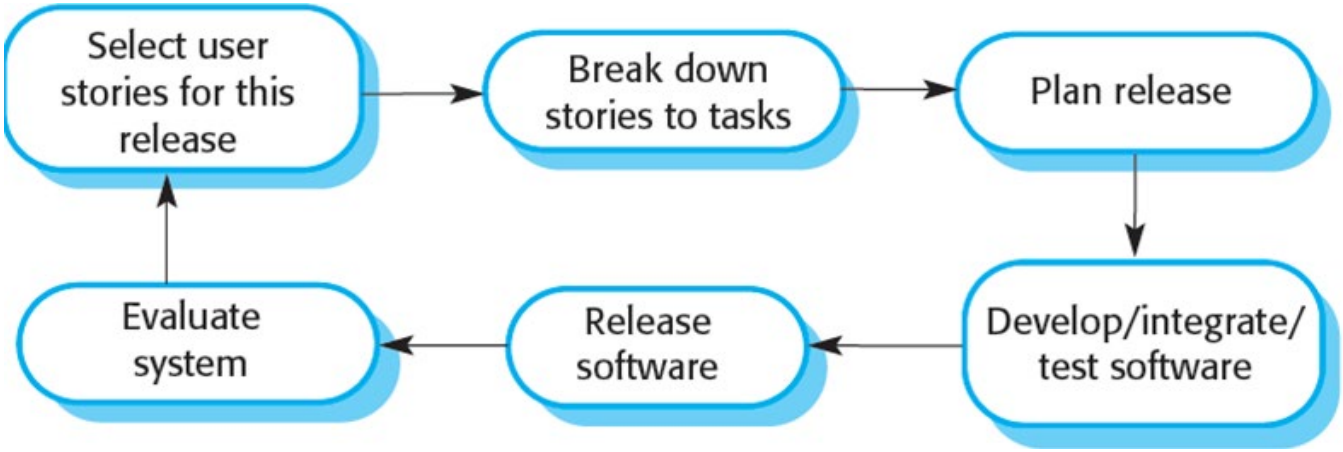
XP 프로세스(계속)

XP에서의 12가지 연습

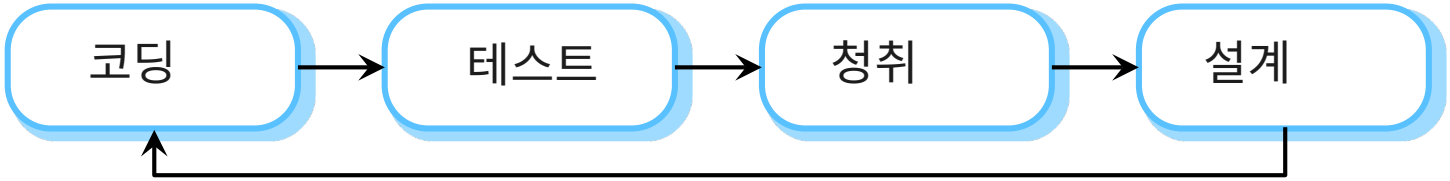
- 계획 과정
- 소규모 릴리스
- 은유
- 심플한 디자인
- 지속적인 테스트
- 리팩토링
- 페어 프로그래밍
- 집단 코드 소유권
- 지속적인 통합
- 40시간 주
- 현장 고객
- 코딩 표준

XP 프로세스 - 개발 주기

릴리스 주기



개발(엔지니어링) 사이클



애자일 방법 – 스크럼

일반적인 애자일 방법이지만 초점은 다음과 같습니다.반복적인 관리에 관하여
개발특정한 애자일 관행이라기보다는.

Jeff Sutherland와 Ken Schwaber의 저서 [Schwaber & Beedle 2002] Scrum 방법은…

- 에이피드백 중심의 경험적 접근 방식 이는 모든 경험적 프로세스 제어와 마찬가지로 3가지 기둥에 의해 뒷받침됩니다.
  투명성, 검사 및 적응.
- Scrum 프레임워크 내의 모든 작업은 다음과 같아야 합니다.결과에 대한 책임이 있는 사람들에게 보여짐 : 프로세스, 작업 흐름, 진행 상황 등
- 이러한 사항을 가시화하려면 Scrum 팀이 다음 작업을 수행해야 합니다. 자주 검사하다 개발 중인 제품과 팀의 작업 능률.

애자일 방법 – 스크럼

스크럼에는 세 가지 단계가 있습니다.

- 초기 단계는 개요입니다계획 단계 프로젝트의 일반적인 목표를 확립하고 소프트웨어 아키텍처를 설계하는 단계입니다.

- 그 다음에는 일련의스프린트 사이클 각 주기가 시스템의 증가를 개발하는 곳입니다.
- 프로젝트폐쇄 단계 프로젝트를 마무리하고, 시스템 도움말 프레임, 사용자 매뉴얼 등 필수 문서를 완성하고, 프로젝트에서 얻은 교훈을 평가합니다.

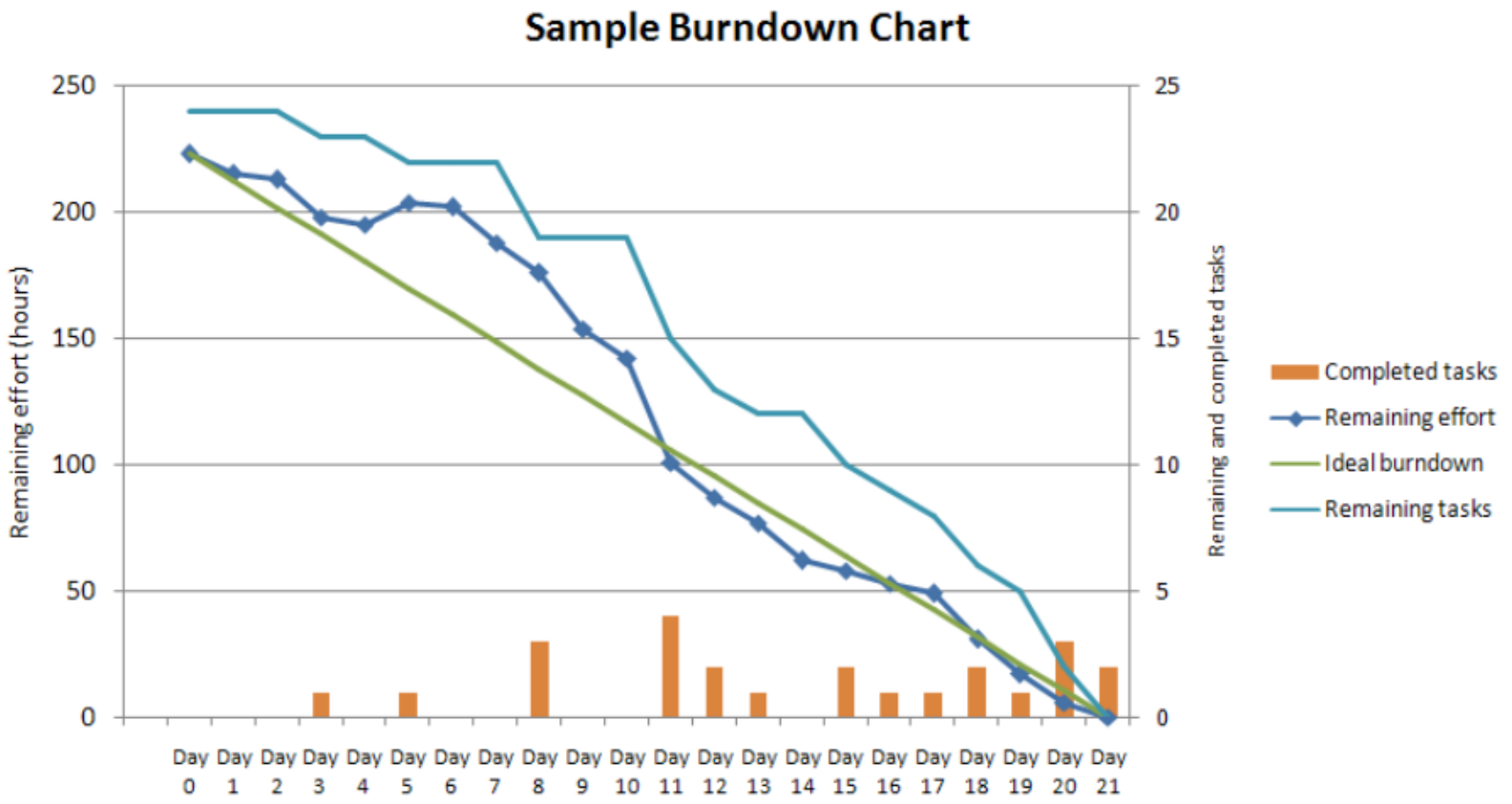
애자일 방법 – 스크럼

유물

- **제품 백로그**
  :Scrum 팀이 유지 관리하는 요구 사항의 정렬된 목록
  제품
- **스프린트 백로그**
  : 개발팀이 다음에 처리해야 할 작업 목록
  스프린트.
- **제품 증가(또는잠재적으로 배송 가능한 증가분, PSI)** : 스프린트 동안 완료된 모든 제품 백로그 항목의 합계
  이전 스프린트의 모든 작업과 통합되었습니다.
- **번다운 차트**
  : 스프린트에서 남은 작업을 보여주는 공개 차트
  매일 업데이트되는 백로그.
  : 번업 차트 :릴리스에 대한 진행 상황을 추적하는 방법

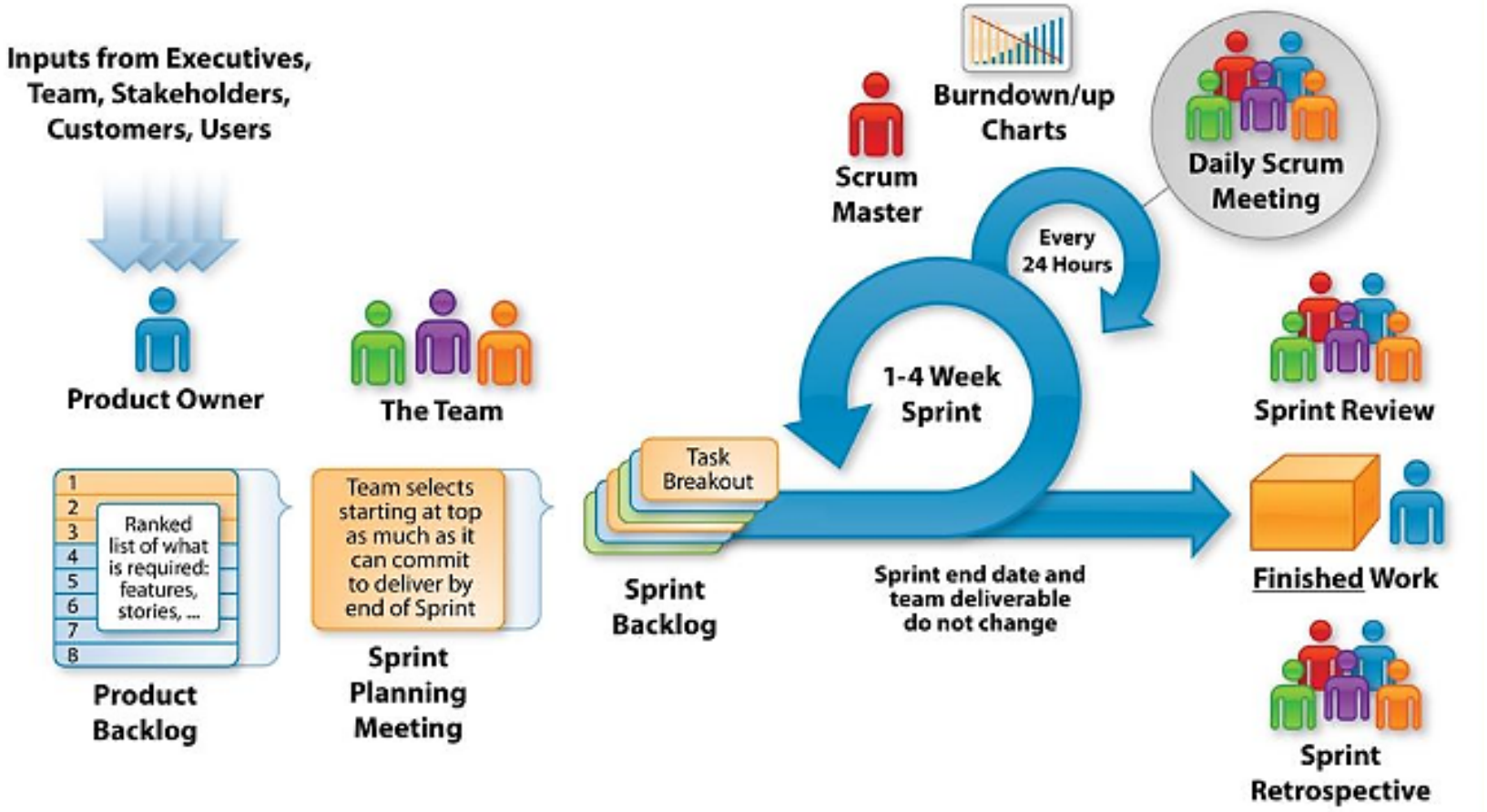
애자일 방법 – 스크럼

버른다운 차트의 예

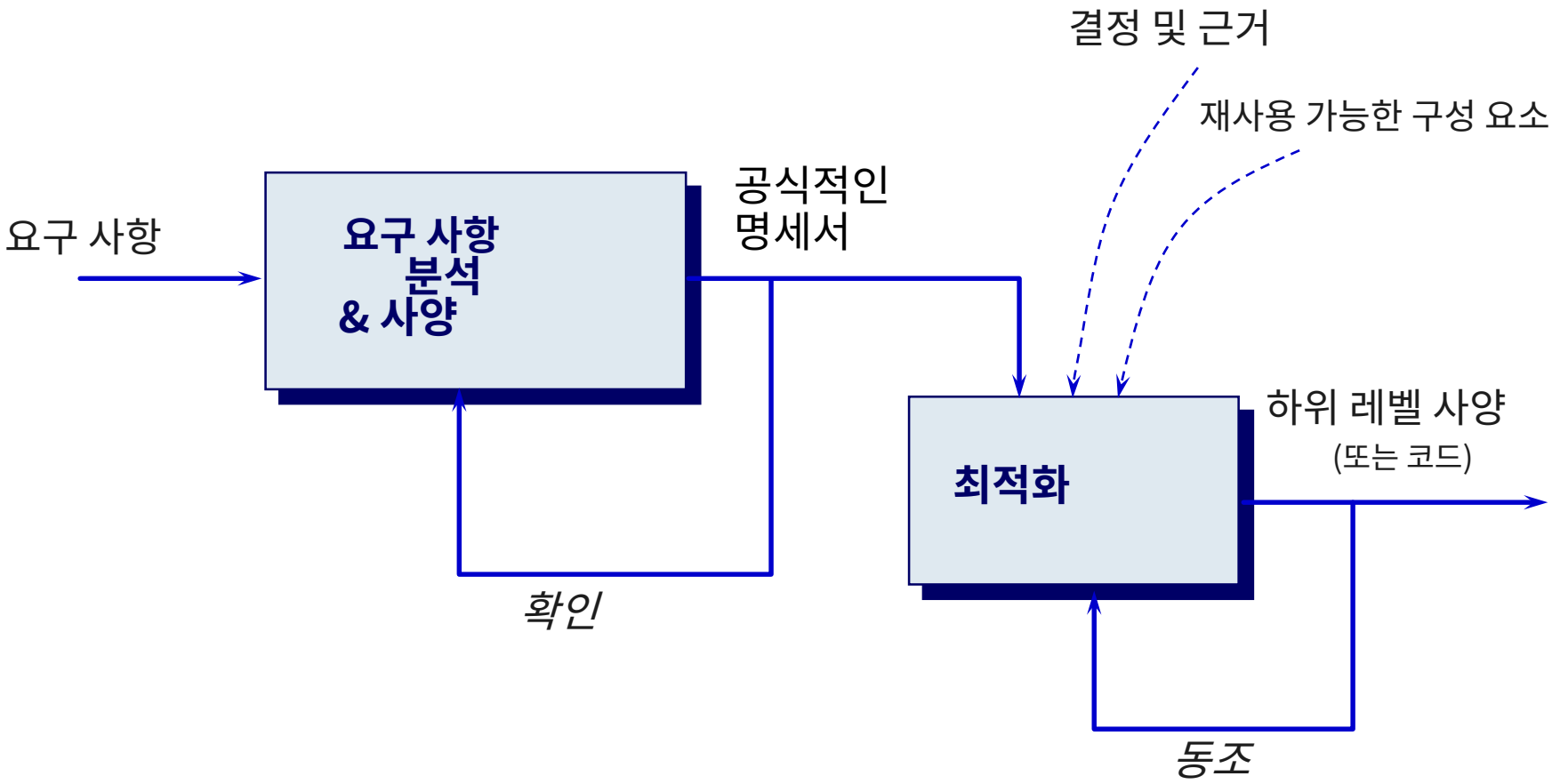


애자일 방법 – 스크럼

스크럼 프레임워크



변환 모델


결정 및 근거
재사용 가능한 구성 요소
요구 사항
요구 사항
분석
& 사양
공식적인
명세서
하위 레벨 사양
(또는 코드)
최적화
확인
동조

변환 모델(계속)

형식적 사양에 근거함

-Z, PetriNet, StateCharts, SDL과 같은…

사양을 점차적으로 변환하는 일련의 단계로 간주됩니다.
구현

수동 및 자동으로

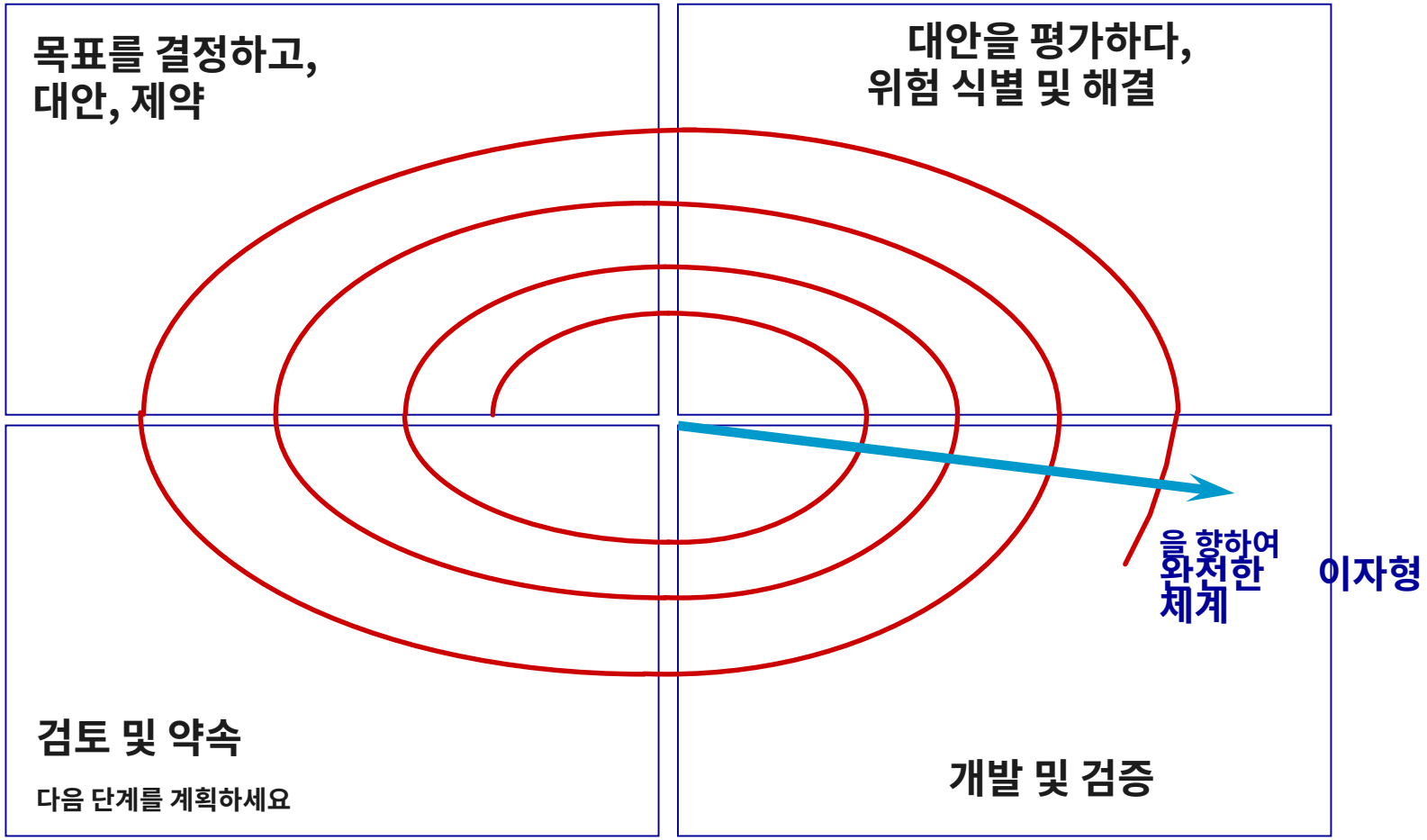
연구 중심적 접근 방식 중 하나(아마도) 프로그램

정확성 증명 문제에 사용됨

-전문 지식이 필요합니다

-산업용으로 좁은 적용범위

나선형 모델


목표를 결정하고,
대안, 제약
대안을 평가하다,
위험 식별 및 해결
검토 및 약속
다음 단계를 계획하세요
개발 및 검증
을 향하여
와처한
체계
이자형

나선형 모델 (계속)

B. Boehm 지음

메타 모델

-소프트웨어 생산 프로세스를 설계하기 위한 프레임워크 제공

-현재 프로젝트의 위험 수준에 따라 안내됨

고위험 문제를 식별하고 제거하는 데 중점을 둡니다.
신중한 프로세스 설계

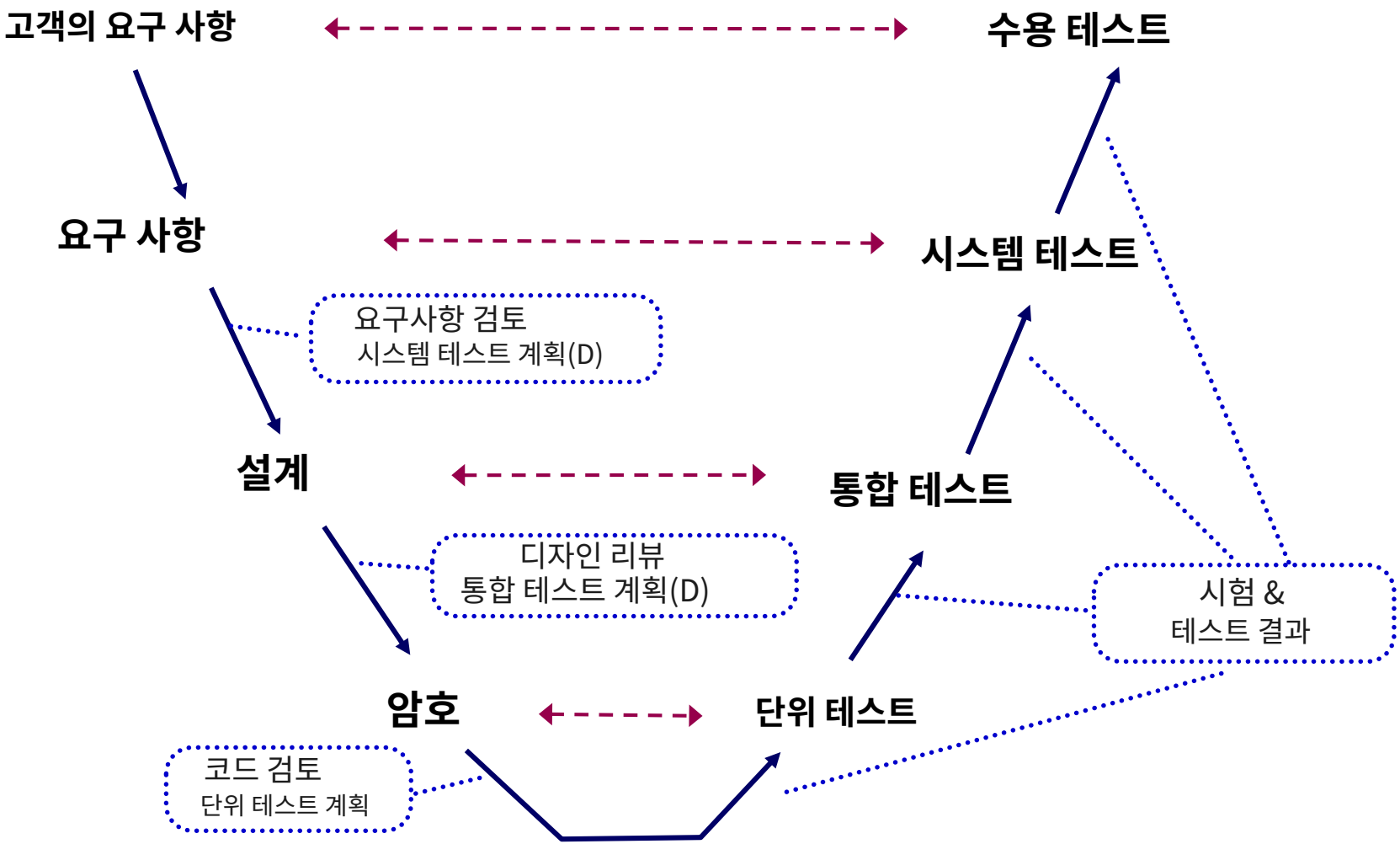
문제들

-각 나선형의 위험 분석으로 인해 비용이 너무 많이 들 수 있습니다.

-적용범위의 제한

-대규모 소프트웨어 개발에만 사용 가능

V 모델


고객의 요구 사항
수용 테스트
요구 사항
시스템 테스트
요구사항 검토
시스템 테스트 계획(D)
설계
통합 테스트
디자인 리뷰
통합 테스트 계획(D)
시험 &
테스트 결과
암호
단위 테스트
코드 검토
단위 테스트 계획

V 모델(계속)

검증 및 확인 모델로 알려짐

폭포수 모델의 확장이며 연관성을 기반으로 합니다.
각 해당 개발 단계에 대한 테스트 단계

장점(+)과 단점(-)

-+ : 매우 엄격한 모델이며 각 단계가 한 번에 하나씩 완료됩니다.

-+ : 요구 사항이 잘 파악된 소규모 프로젝트에 적합합니다.

-+ : 간단하고 이해하기 쉬우며 사용하기 쉽습니다.

-– :요구 사항이 변경될 위험이 중간에서 높은 프로젝트에는 적합하지 않습니다.

-– :복잡한 모델에는 적합하지 않습니다.및 객체 지향 프로젝트.

CBSE 프로세스

구성 요소 기반 소프트웨어 엔지니어링 프로세스는 1968년

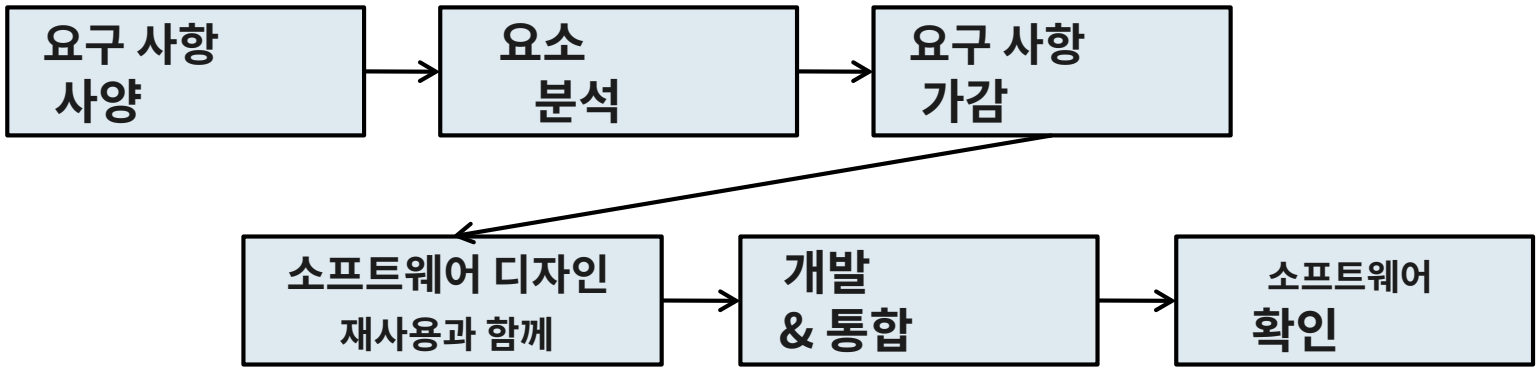
Douglas Mcllroy와 함께 처음 알려졌습니다.

-대량 생산된 소프트웨어 구성 요소

Brad Cox가 1986년에 제시한 소프트웨어 구성 요소의 현대적 개념

-소프트웨어 IC

CBSE 프로세스를 통한 개발



데브옵스 =개발도피 +오피에레이션

민첩성, 협업 및
IT 및 개발팀 프로세스 자동화.

전통적으로 소프트웨어 개발

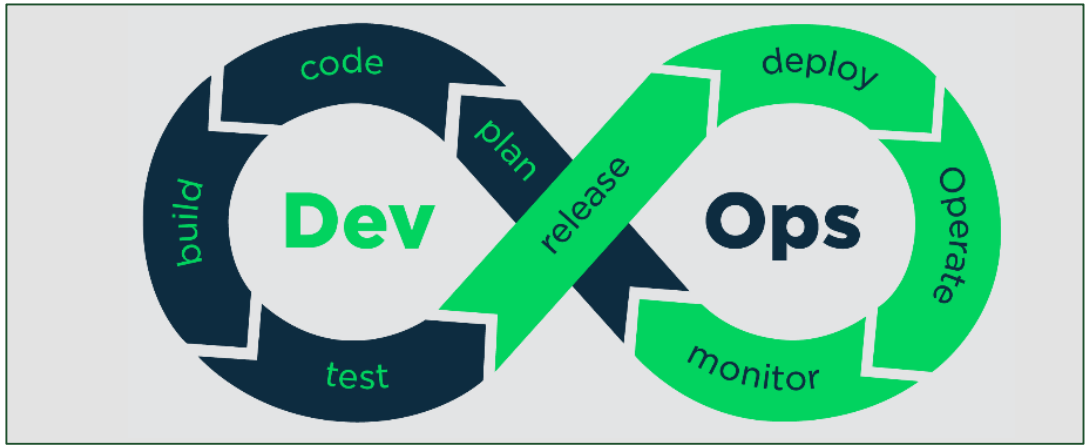
- 사일로 접근 방식
- 각자의 팀과 프로세스에 따라 독립적으로 작업합니다.
- 오해, 부적절한 정렬, 생산 지연이 만연한 환경("작전실").

데브옵스 목표

- IT 운영과 개발 간의 격차를 메우다
- 소통과 협업을 개선하다
- 보다 원활한 프로세스를 구축하고 전략과 목표를 일치시킵니다.
- 더욱 빠르고 효율적인 배송.

DevOps란 무엇인가요?

DevOps의 개발-운영 라이프사이클



SUSE에서

- 지속적인 통합
- 지속적인 배달
- 지속적인 배포
- 마이크로 서비스 아키텍처
- 코드 기반 인프라
- 모니터링 및 로깅

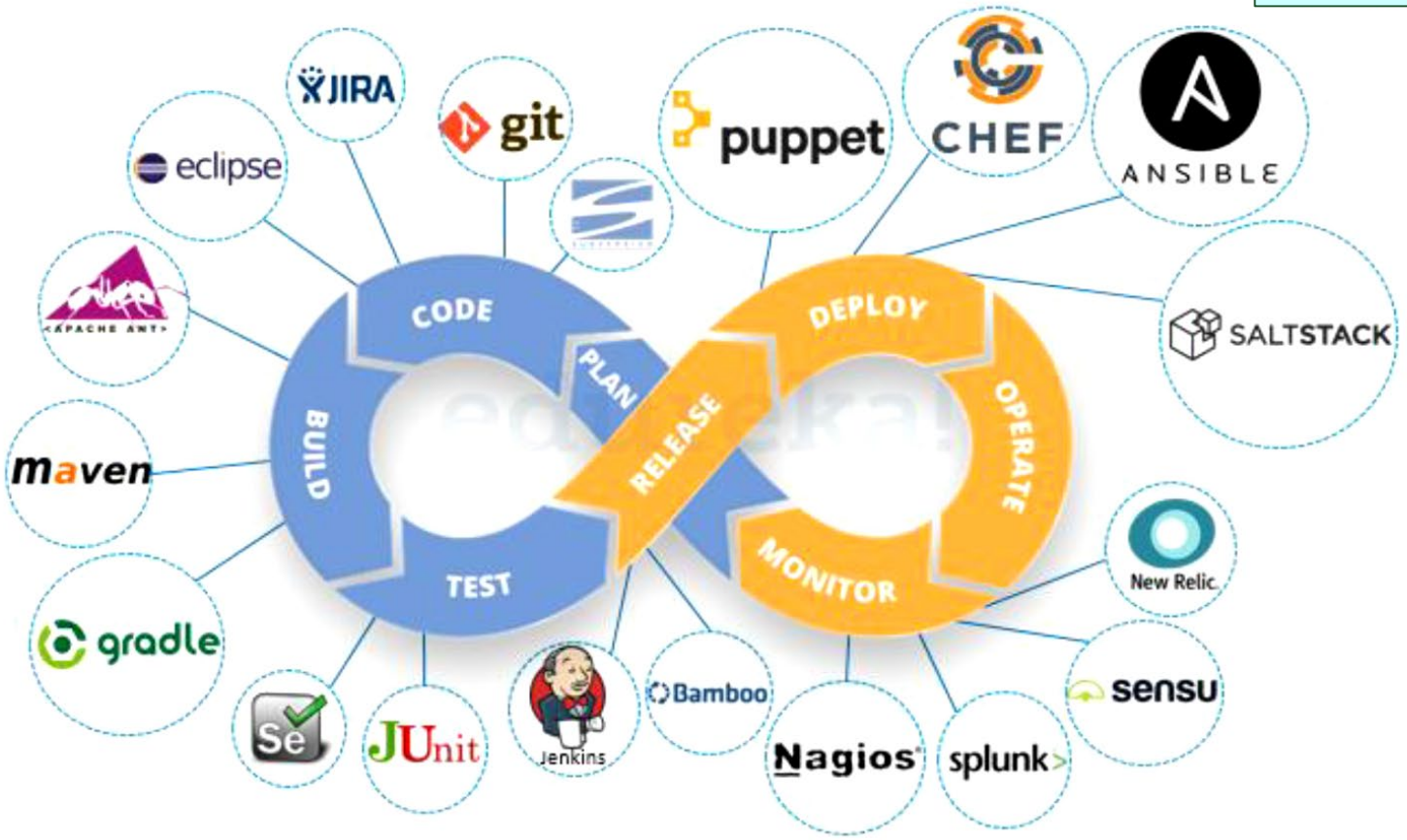
이익

- 빠른 개발 및 빠른 릴리스
- 테스트 자동화
- 빠르고 쉬운 업그레이드
- 협력 강화
- 보안 프로세스

DevOps의 툴체인

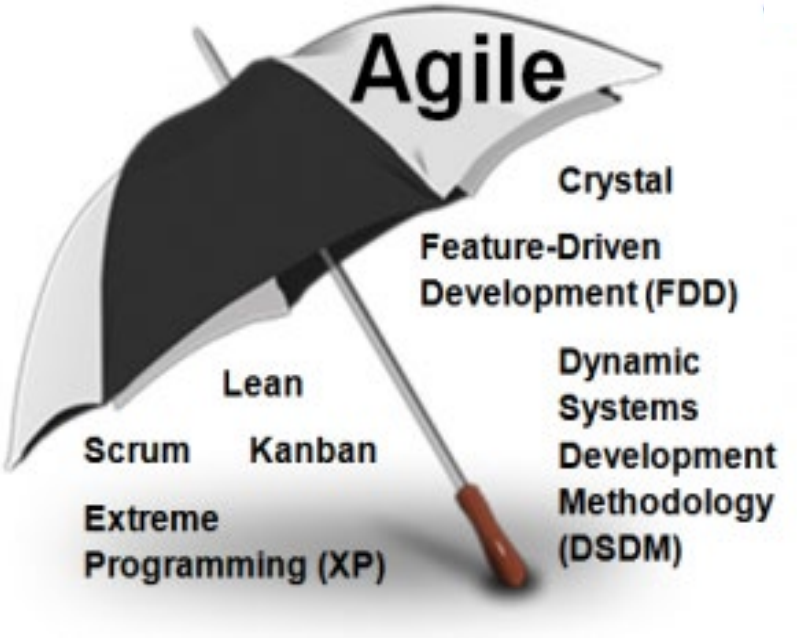
자동화는 핵심 중 하나입니다.

예



기타 프로세스 모델

그리고 더 많은 것들… ?!



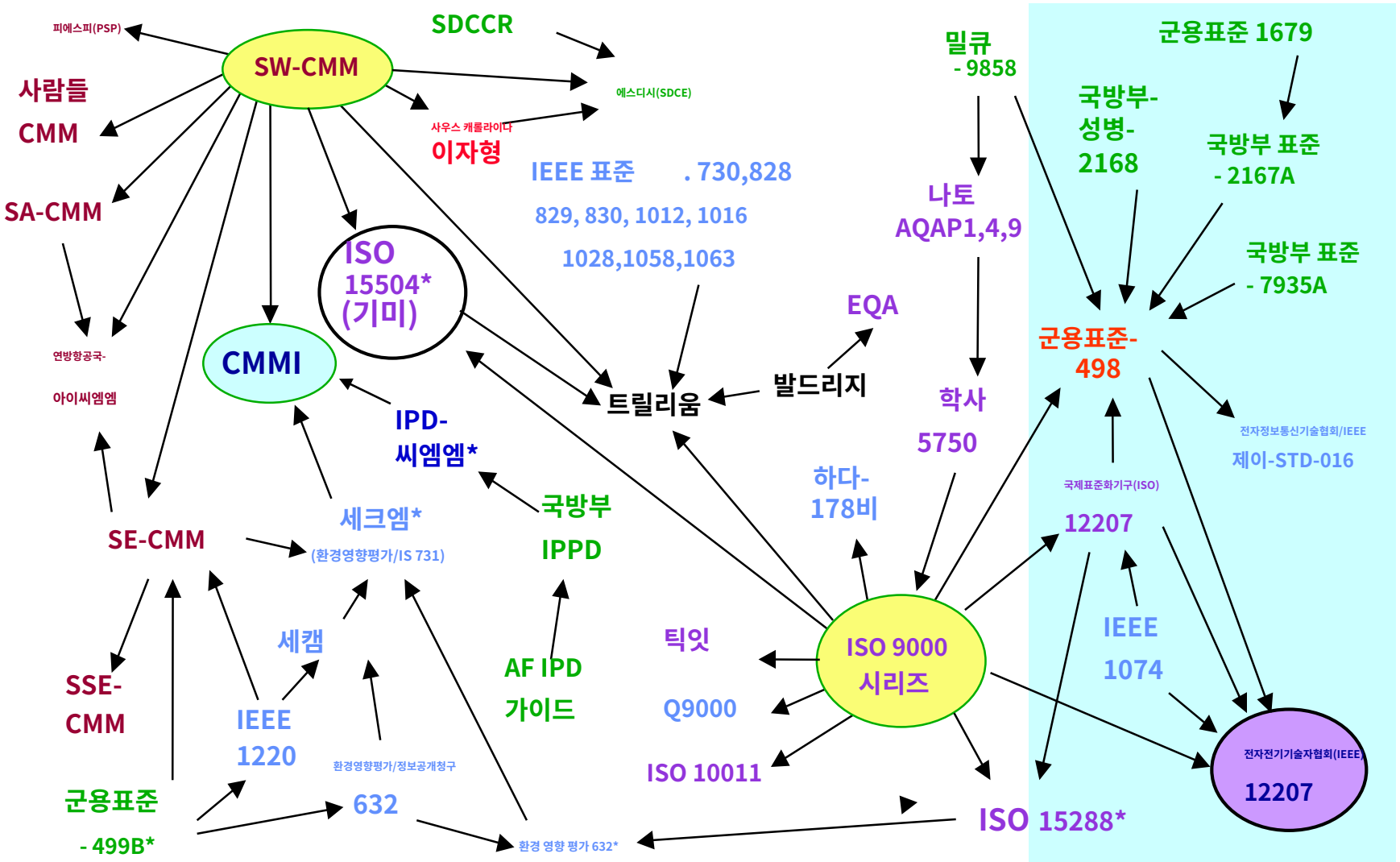
프로젝트 특성별 SDLC

형질	폭포	프로토타입	나선	증분형	반복적	기민한
대판			-	-	-	
많은 위험		-	-	-		
모호한 요구 사항		-		-		-
장기적으로			-	-	-	
충분한 예산			-	-		
쉬운 기술	-					-
높은 정확성		-	-	-		-
고객 참여						-

주요 출력 문서

단계	서류	구성 요소	비고
프로젝트 계획	**프로젝트 경영 계획**	- 관리 문제(비용, 일정, 리소스 등) - 품질 관리 계획 - 구성 관리 계획	
요구 사항 모임	**RDD(요청) (요구사항 설명 문서)**	- 시스템 설명 - 기능적 요구 사항 - 비기능적 요구 사항	
요구 사항 분석	**SRS** **(소프트웨어 필요) 사양)**	- 기능 분석 모델 - 데이터 모델	
설계	**에스디디 (소프트웨어 설계 설명)**	- 예비 설계 - 세부 디자인(데이터, 인터페이스 등) - 배치	분리되어 어떤 경우.
구현	**소스 코드 목록**	- 코드	
테스트	**테스트 계획** **테스트 결과**	- 테스트 목표(종료 기준), 테스트 케이스 - 테스트 실행 보고서	

소프트웨어 프로세스에 대한 표준



www.software.org/quagmire

ISO/IEC/IEEE 12207

목적

-소프트웨어 수명 주기에 대한 공통 프레임워크를 구축하려면 다음을 수행하세요.

- 소프트웨어를 취득, 공급, 개발, 운영 및 유지 관리합니다.
- 프레임워크를 관리, 제어 및 개선하려면

역사

-1988년 6월에 제안됨

-4개의 작업 초안, 2개의 위원회 초안, 1개의 DIS

-6년 이상, 17000인시 투자

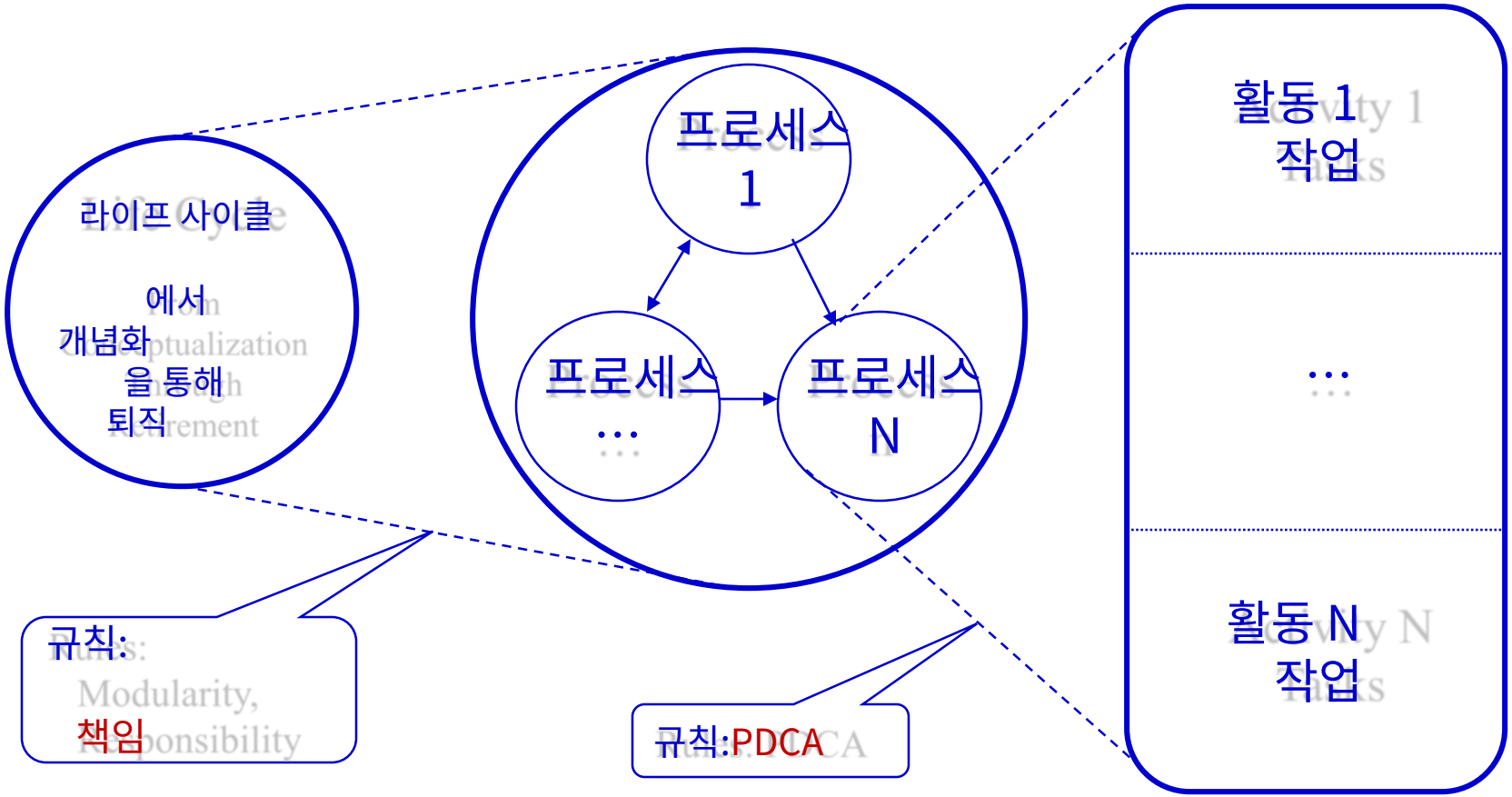
-1995년 8월 1일 출판

-2017년 11월 새 버전

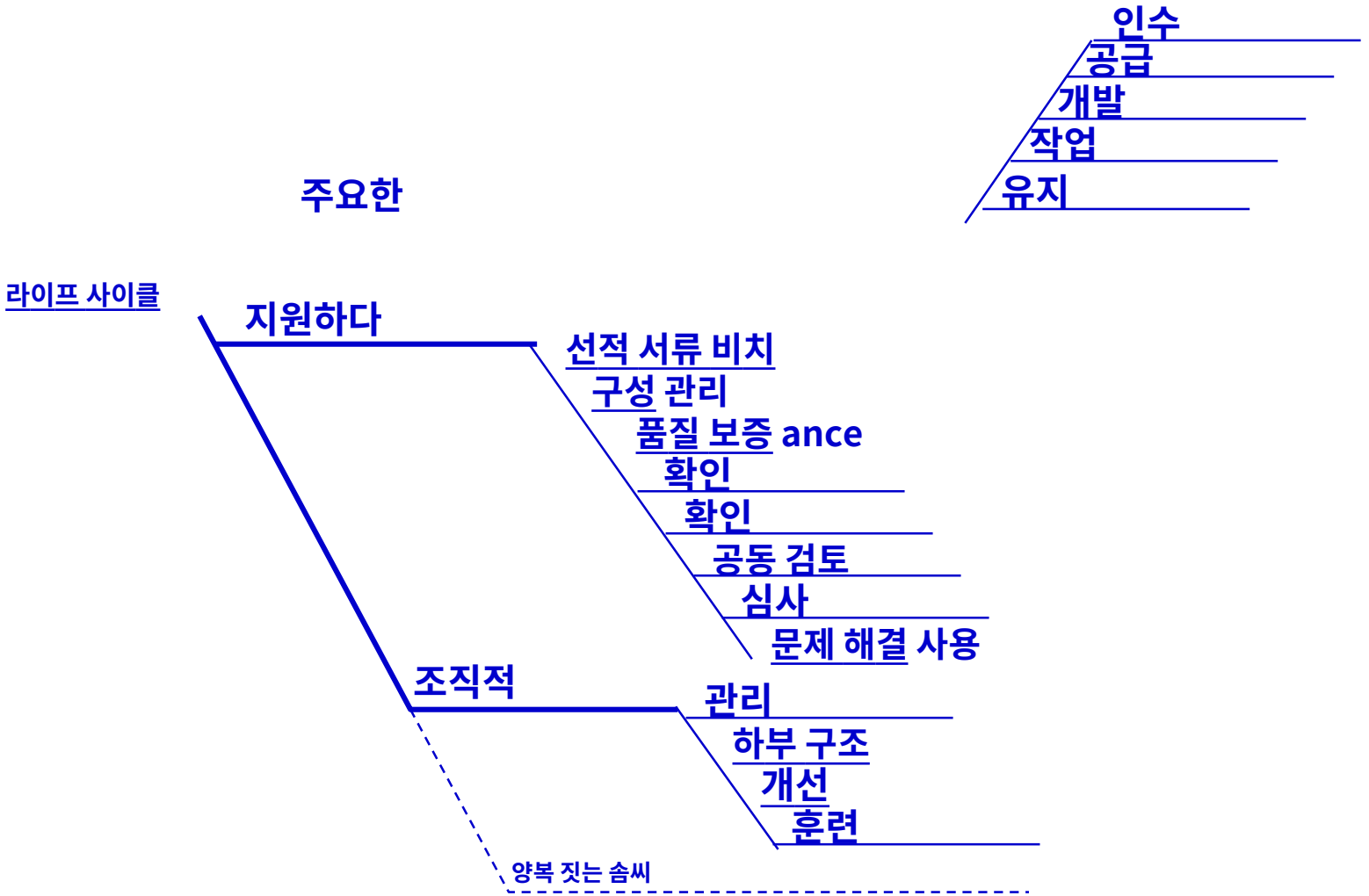
참가자들

-호주, 캐나다, 덴마크, 핀란드, 프랑스, 독일, 아일랜드, 이탈리아, 일본, 한국, 네덜란드, 스페인, 스웨덴, 영국, 미국

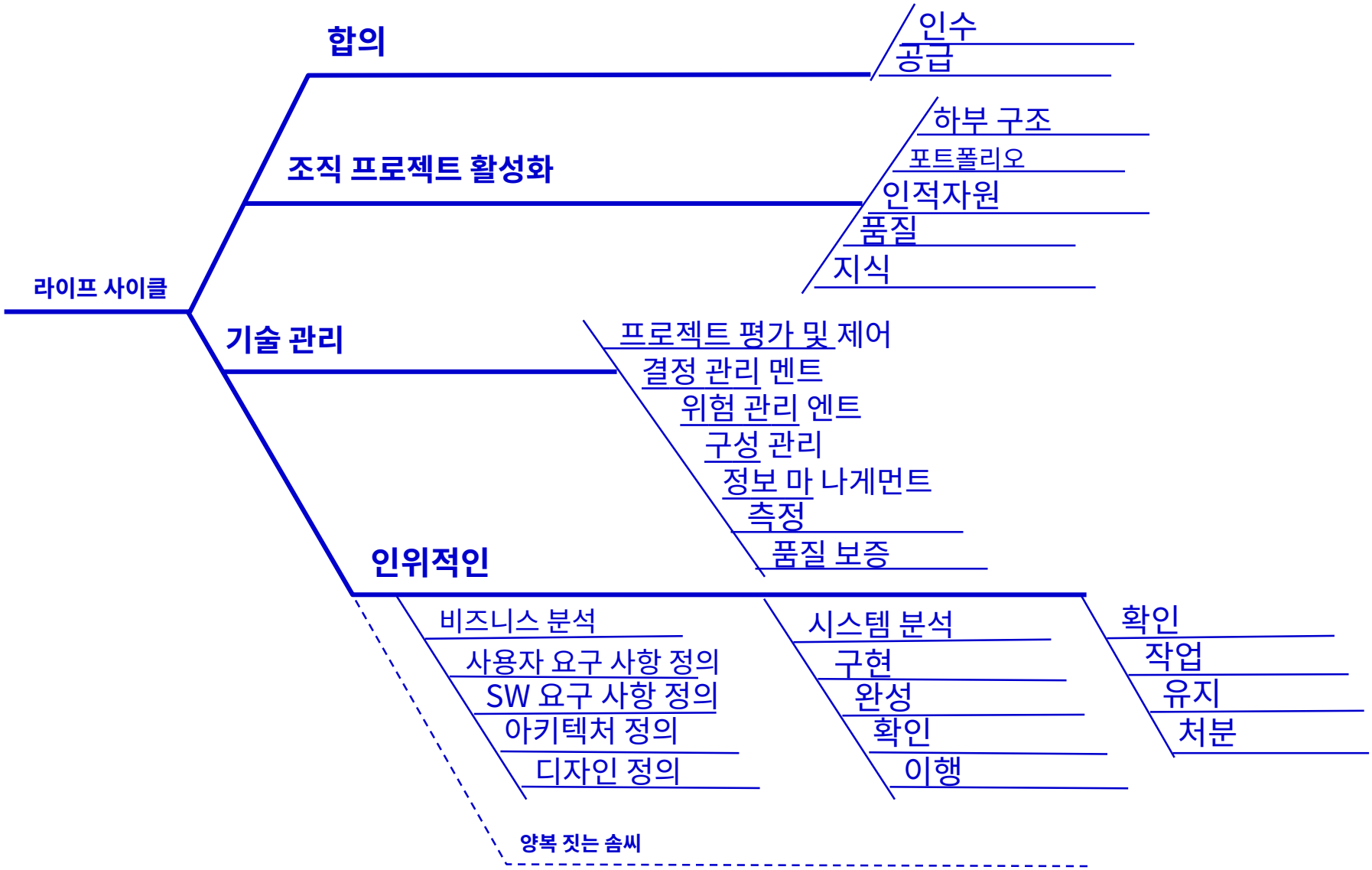
기본 개념 - 아키텍처


라이프 사이클
에서
개념화
을 통해
퇴직
프로세스
1
프로세스
…
프로세스
N
활동 1
작업
…
활동 N
작업
규칙:
책임
규칙:PDCA

기본 개념 - 프로세스(1995)



기본 개념 - 프로세스(2017)



레거시 소프트웨어 처리

동기 부여

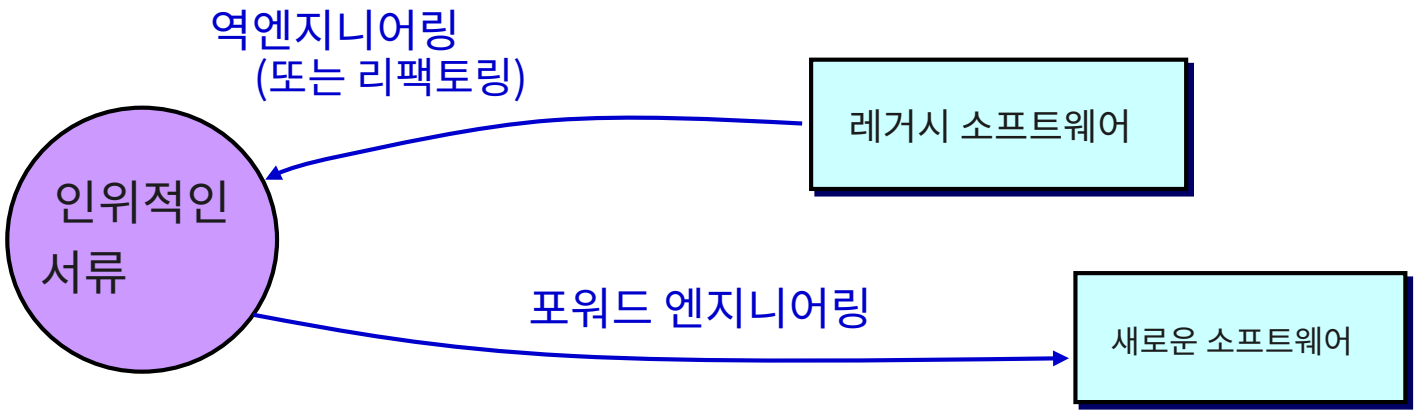
-새로운 소프트웨어를 처음부터 개발하는 것은 불가능합니다.

– 기존 소프트웨어 개발에 막대한 투자

-유산: 폐쇄하기 전에 매우 신중하게 보존해야 할 자산

리엔지니어링

-기존 시스템이 변형되어 새로운 형태로 재구성되는 과정



요약 및 토론

소프트웨어(생산) 프로세스 모델

- 폭포
- 진화적(빠른 프로토타입 제작, 증분적)
- 변환
- 나선형 모델
- 애자일 모델
- DevOps 모델, ...

왜 이러한 프로세스 모델이 필요할까요?

프로세스 모델의 차이점은 무엇입니까?
그리고 방법론은?