델마당 연합세미나 2008
소프트웨어아키텍쳐(입문)

Ver 1.3 (2008/10)

꿈꾸는자(zetlos)

http://zetlos.tistory.com

우리들그룹

온라인총괄이사 신현묵

http://www.wooridul.co.kr

꿈꾸는자...

블로그 : http://zetlos.tistory.com

예전경력
- 델마당 2대 회장,
- 하이텔 비주얼파워툴 동호회 3,4대 대표시삽
- OMG Member

책 관련
- 델파이4의 모든것 , 애니메이션 잡지 4권 기획

현재 하는일
- 우리들그룹(계열사 17개)의 On-Line/IT관련 업무를 총괄하고 있음.

소프트웨어개발이란?

단순히 프로그램 제작? 전체 단계를 모두?

도공 陶工

[명사] 옹기장이

= 옹기장이

[명사] 옹기 만드는 일을
업으로 하는 사람

예술 작품, 작가의 만족, 고객은 어디에?

노산진이 이야기하는 도자기...

'인간 국보'지정 거부한
일본 도예가의 삶..

– 기타오지 노산진

도자기는 **그림**이 아니다.
그림은 혼자만이 할 수 있지만, 도자기는 흙을 정제하는 자,
유약을 만드는 자, 물레 차는 자, 그림을 그리는 자,
조각을 하는 자 그리고 불 때는 자 이 모든것이 어우러져 나오는
협동예술이다.
조선의 도자기, 중국의 도자기, 아리타의 백자도 그렇다.

도예가가 도자기의 이 모든 공정을 혼자만 한다면
보다 나은 도자기 세계로 나아가기 힘들다.

나는 이 모든 것을 조합하여 한데 묶어 최종적으로 가장 좋은
도자기를 만들기 위해 노력하는 자이다.

아키텍처는 컴포넌트로 구체화된 시스템의 기본적인 조직이며,
환경에 대한 관계이며, 디자인과 진화를 이끄는 원리이다. [IEEE1471]

소프트웨어/개발

사용자가 원하는 서비스를 제공하기 위하여
정보기술을 활용하여 서비스를 구축하는 것!.

소프트웨어

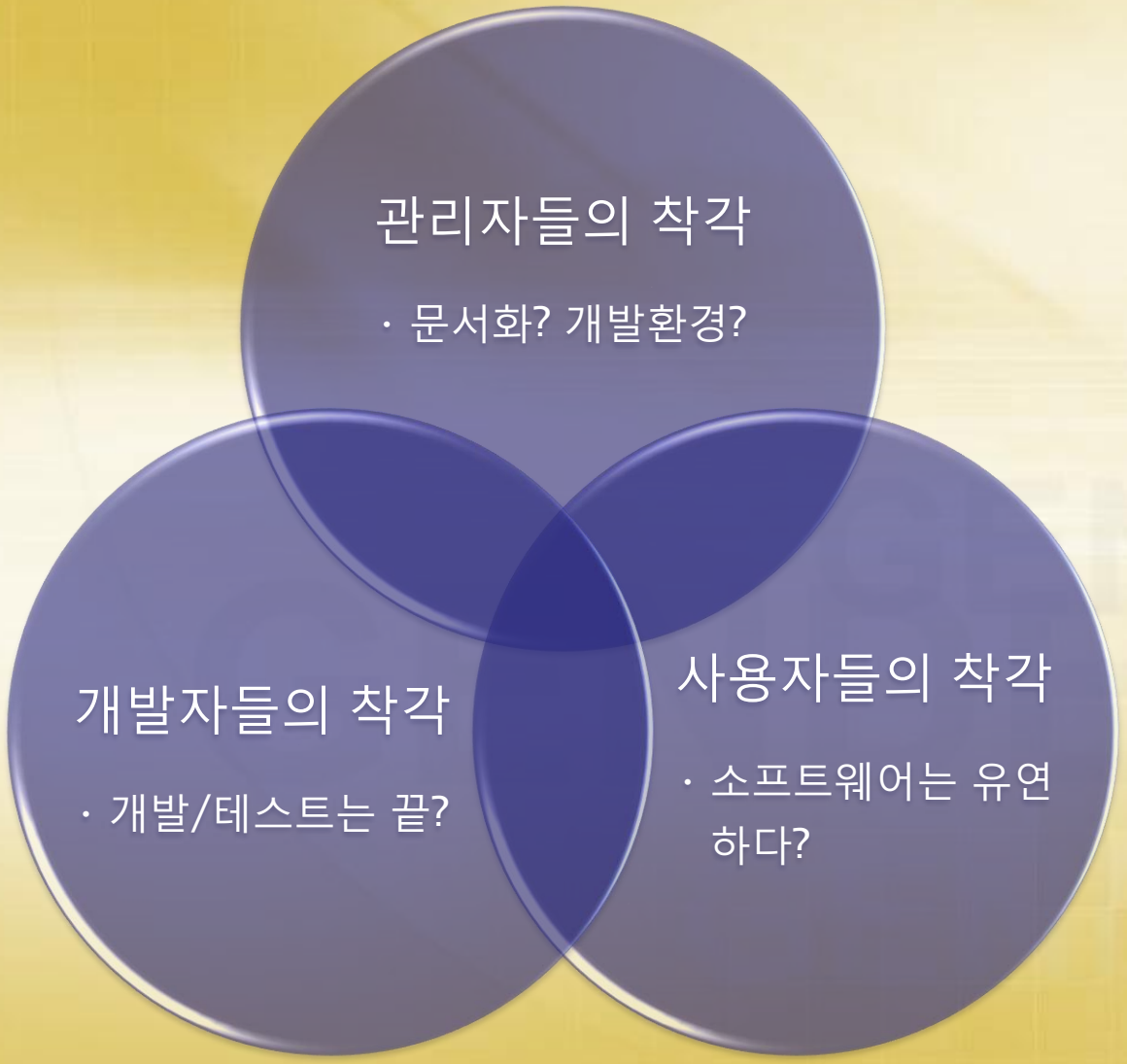
서비스

개발

소프트웨어 공학

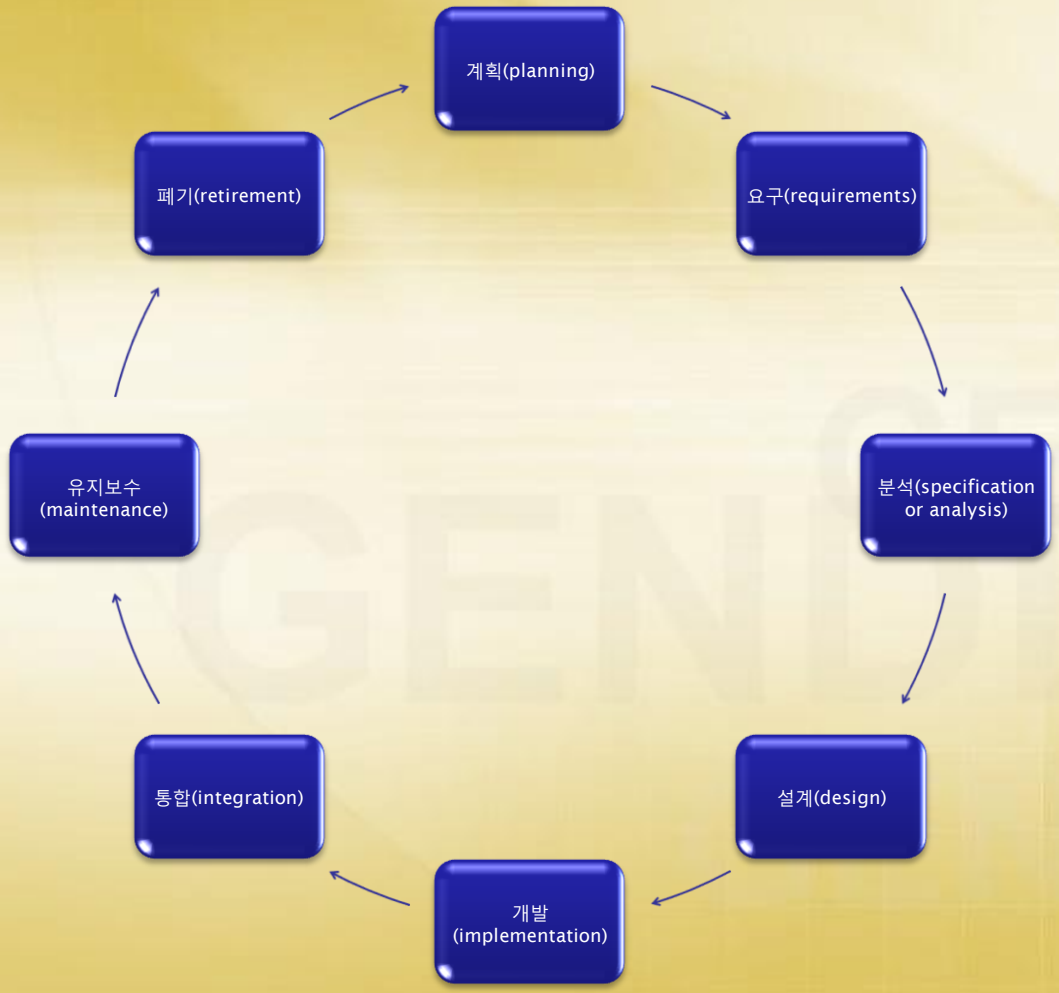

견고성
비마모성
상품성
개발
(비제조성)
Dynamic
behavior
개발
(비조립성)
tangibility
Software
개발
(비과학성)

잘못된 의식구조

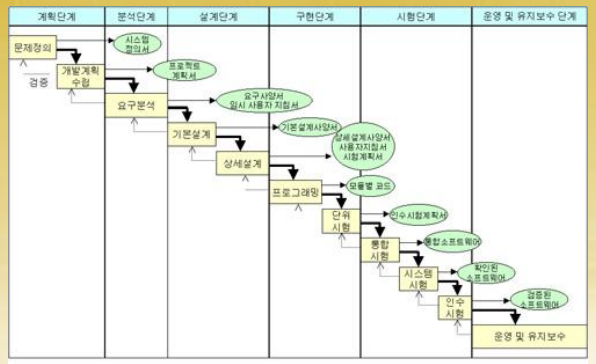


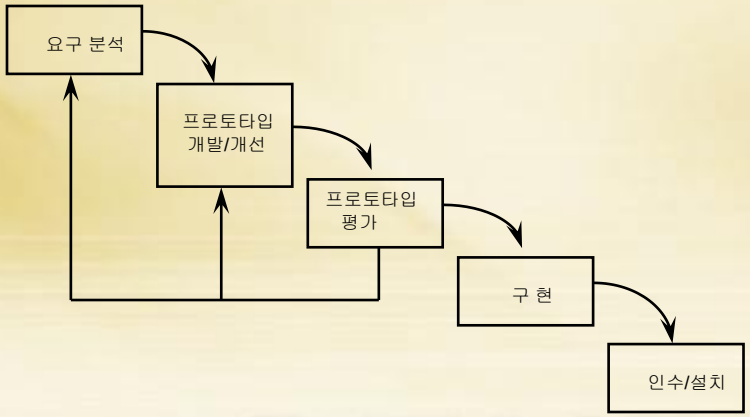
소프트웨어 공학

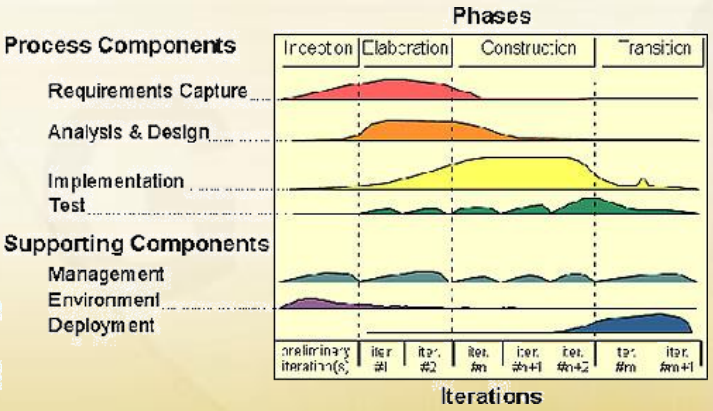
소프트웨어 생명주기 모델

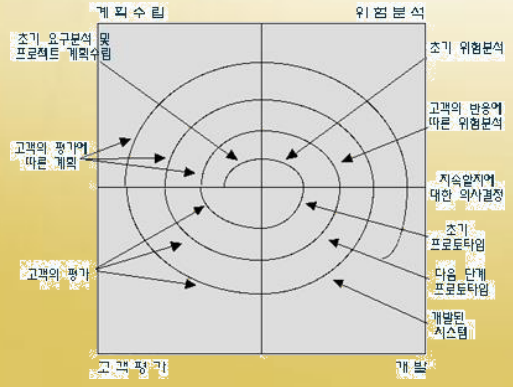


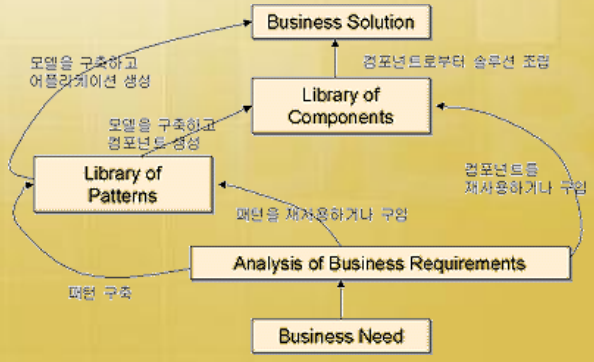
개발모델


계획단계
분석단계
설계단계
구현단계
시험단계
운영 및 유지보수 단계
문제정의
개발계획 수립
검증
시스템 정의서
프로젝트 계획서
요구분석
요구사양서 일시 사용자 지침서
기본설계
기본설계사양서
상세설계
상세설계사양서 사용자지침서 시험계획서
프로그래밍
모듈별 코드
단위 시험
인수시험계획서
통합 시험
통합소프트웨어
시스템 시험
확인된 소프트웨어
인수 시험
검증된 소프트웨어
운영 및 유지보수


요구 분석
프로토타입 개발/개선
프로토타입 평가
구 현
인수/설치

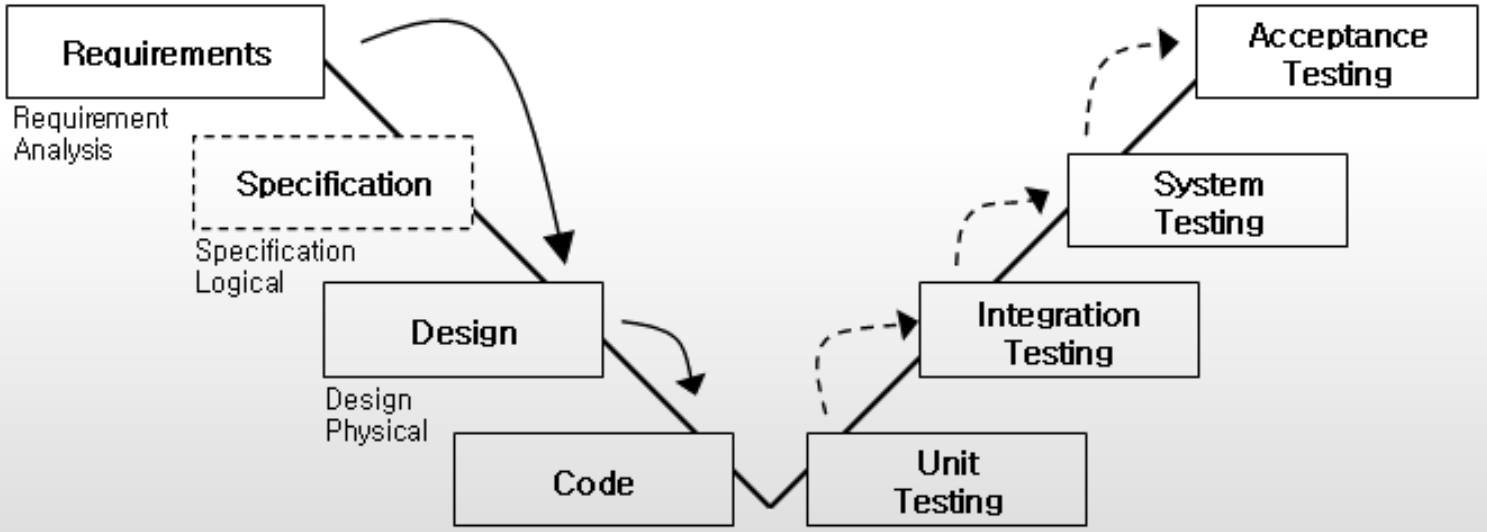

Phases
Process Components
Inception
Elaboration
Construction
Transition
Requirements Capture
Analysis & Design
Implementation
Test
Supporting Components
Management
Environment
Deployment
Iterations


계획수립
위험분석
초기 요구분석 및 프로젝트 계획수립
초기 위험분석
고객의 반응에 따른 위험분석
고객의 평가에 따른 계획
지속여부에 대한 의사결정
초기 프로토타입
고객의 평가
다음 단계 프로토타입
개발된 시스템
고객평가
개발

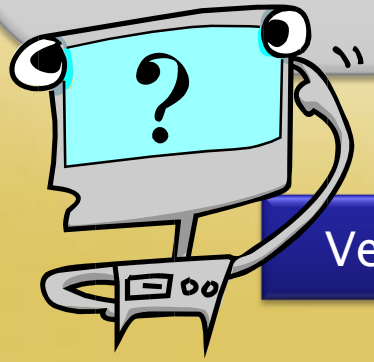

Business Solution
모델을 구축하고 어플리케이션 생성
컴포넌트로부터 솔루션 조립
Library of Components
모델을 구축하고 컴포넌트 생성
Library of Patterns
컴포넌트를 재사용하거나 구입
패턴을 재사용하거나 구입
Analysis of Business Requirements
패턴 구축
Business Need


작업절차
도구
작업방법
방법론
관리
산출물
기법

V-model



V - 모델



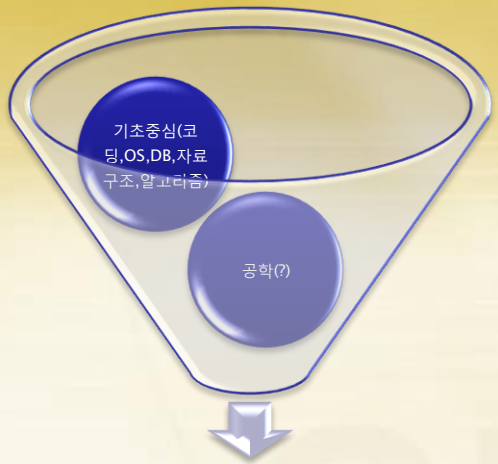
Verification

Validation

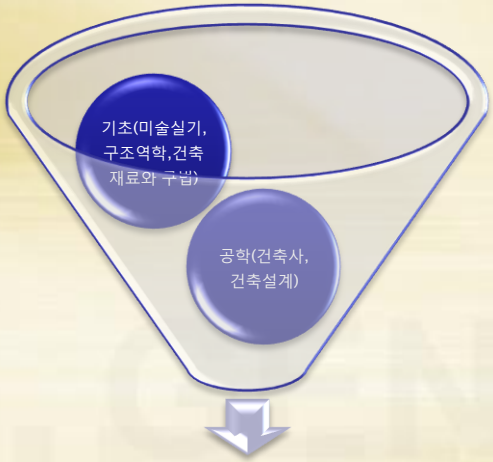
우리는 람보가 아니다?

람보!

IT와 건축



전산학과



건축학과

갑 을 병 정

노가다

- 노가다 [dokata /土方]
  - ‘공사판 노동자’,’노동자’,’막일꾼’,’인부’로 순화
  - 행동과 성질이 거칠고 불량한 사람을 속되게 이르는 말
  - 건축일 등 단순한 노동일을 일컫는 말
  - 특별한 직업이 없는 사람들이 흔히 일당제로 하는 일

- 건축현장 / 소프트웨어 개발
  - 우리는 과연 누구인가?
  - 10년 등짐지었다고 건물을 지을 수 있는가?
    - 지을 수 있다! (빌라 3층은 가능하더라.)
      - 대신. 비올때 마다 세고…
  - 건물이 하나 만들어졌다.
    - 과연 누가 이 건물을 만들었다고 하는가?
    - 과연 누구의 작품인가?

좌우당간!

- 서있는 곳이 다르면 보이는 '곳'이 다르다.

개발자고수와 아키텍트?

코드 구루(Guru)

로직에 대한 코딩의 책임

문제해결능력!

코딩의 달인!

Recipe

조리법/비법/비결/약제 처방전
재료손질법, 일정한 순서, 먹는방법,

레시피를 만드는 요리사!

아키텍트?

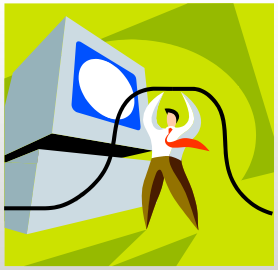
1인 개

소프트웨어 아키텍쳐

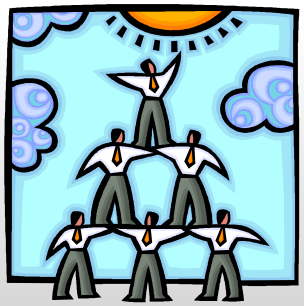
-아키텍트의 역할 -

‘ 소프트웨어 개발과 관련해 기술적인 판단을 내리는 최고책임자’

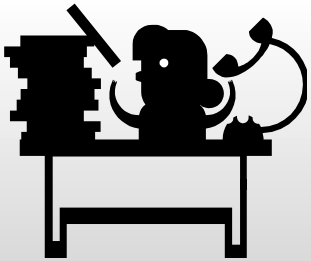
비용



적용기술



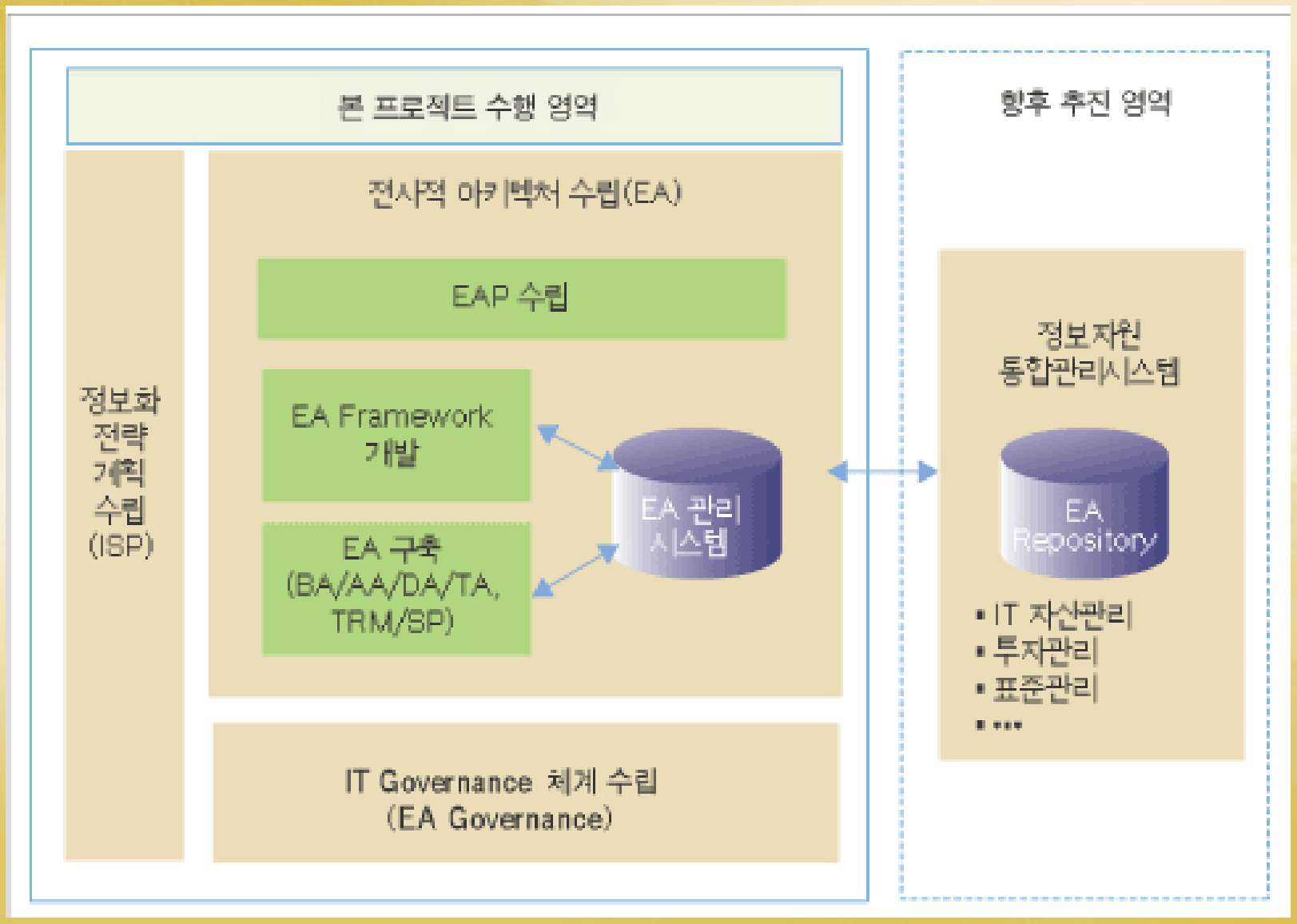
조직의 역량



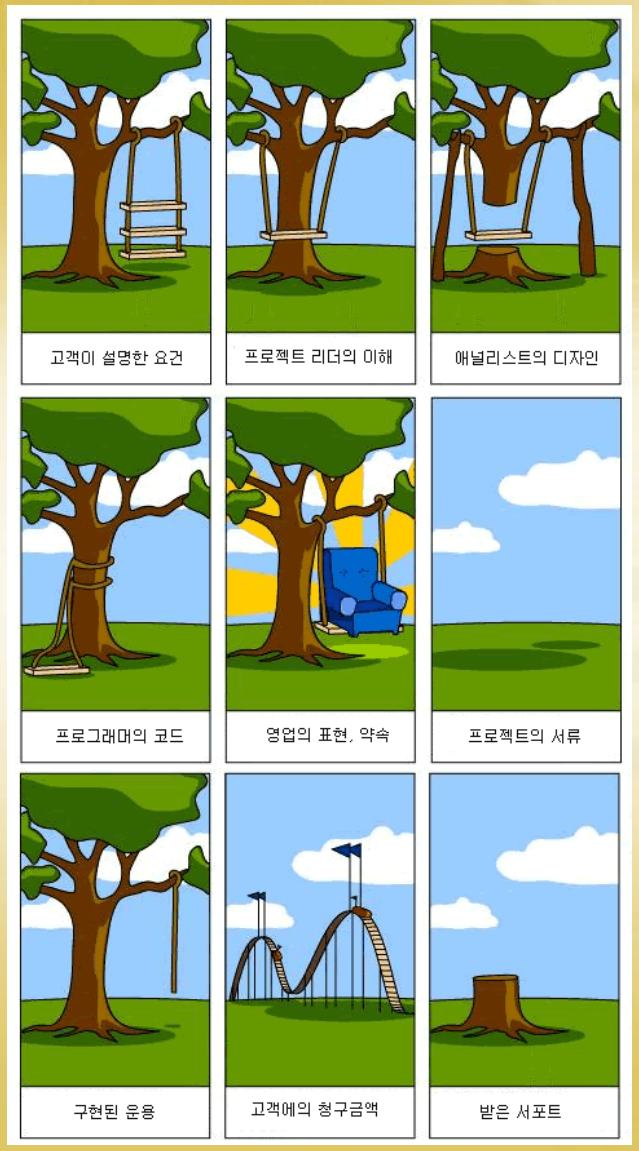
기능의 복잡도

목표

EA


본 프로젝트 수행 영역
향후 추진 영역
정보화 전략 계획 수립 (ISP)
전사적 아키텍처 수립(EA)
EAP 수립
EA Framework 개발
EA 구축 (BA/AA/DA/TA, TRM/SP)
EA 관리 시스템
정보자원 통합관리시스템
EA Repository
• IT 자산관리
• 투자관리
• 표준관리
• ...
IT Governance 체계 수립 (EA Governance)

프로젝트의 현실!

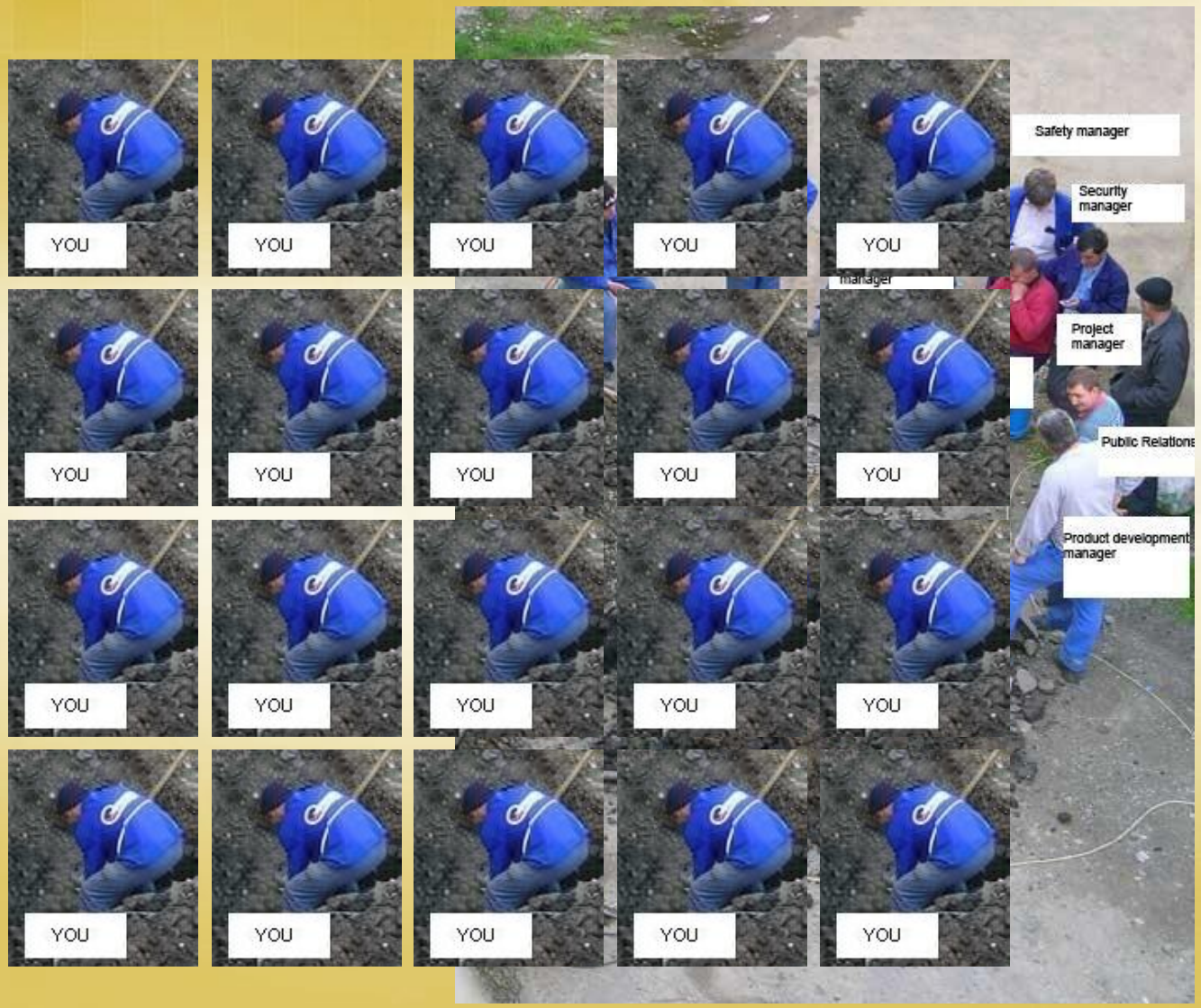

고객이 설명한 요건
프로젝트 리더의 이해
애널리스트의 디자인
프로그래머의 코드
영업의 표현, 약속
프로젝트의 서류
구현된 운용
고객에의 청구금액
받은 서포트


고객이 정말 필요한 것

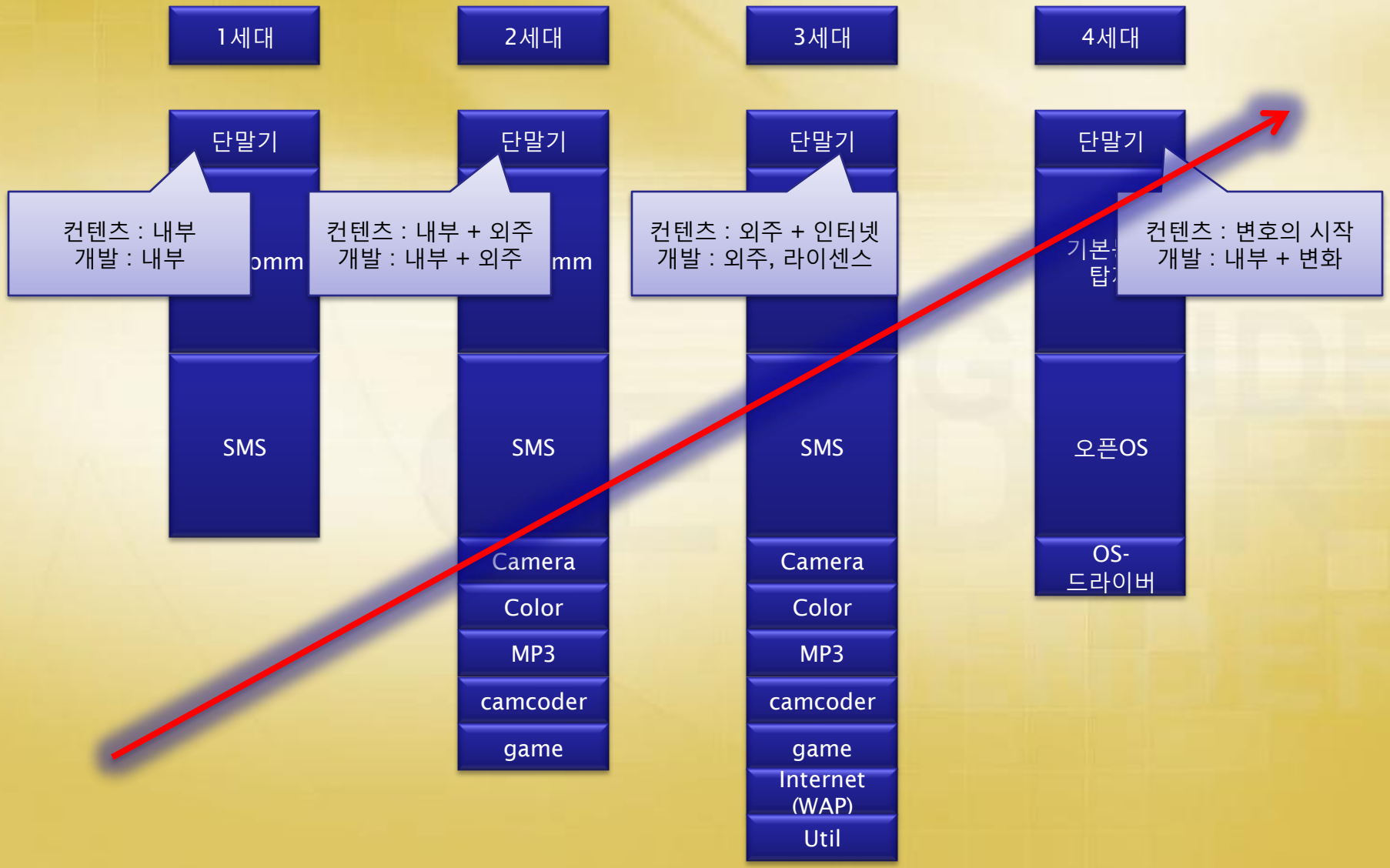
프로젝트의 현실


Logistics manager
Marketing manager
HR manager
Safety manager
Security manager
Communications manager
Project manager
QA/QC manager
Public Relations
Product development manager
YOU

프로젝트의 현실2


YOU
Safety manager
Securtiy manager
Project manager
Public Relations
Product development manager

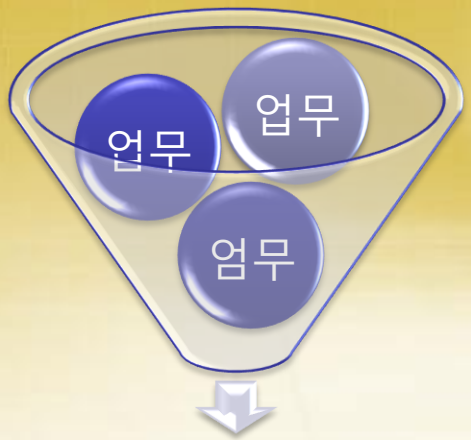
휴대폰 플랫폼의 변화


1세대
2세대
3세대
4세대
단말기
단말기
단말기
단말기
컨텐츠 : 내부
개발 : 내부
컨텐츠 : 내부 + 외주
개발 : 내부 + 외주
컨텐츠 : 외주 + 인터넷
개발 : 외주, 라이센스
컨텐츠 : 변호의 시작
개발 : 내부 + 변화
SMS
SMS
SMS
오픈OS
Camera
Color
MP3
camcoder
game
Camera
Color
MP3
camcoder
game
Internet
(WAP)
Util
OS-
드라이버

노르웨이 어업의 변화

- 국내 어업과 다른점
  - 소규모 어선 -> 개인어부는 거의 없음
  - 거의 신고제 -> 정부에서 라이센스 발급 어업 회사
  - 작은 규모, 수작업 -> 전자동 운행 선박(!)
  - 물류도 적음 -> 독자적 물류 시스템 전세계 커버
  - 사장 사업 -> 2만명의 어민이 세계 10대 수산 대국

소프트웨어 품질 결정



100% 고품질?

사용자 인터페이스 설계 원칙

설계자 관점의 원칙

- 사용자와 작업 중심의 설계
- 기능성 중심의 설계
- 이용자 테스트를 통한 설계 보완
- 사용자 관점에서 설계
- 배우기 쉬운 인터페이스 설계
- 사용자가 작업 수행을 간단, 명료하게 진행하도록 설계
- 적절한 피드백을 제공하는 인터페이스 설계
- 단순 데이터가 아닌 정보를 전달하는 인터페이스 설계

사용자 관점의 원칙

- 배우는데 걸리는 시간
- 작업 실행 속도
- 기억력 (걸린 시간, 사용빈도, 사용법 익히는 속도)
- 사용자의 주관적인 만족도

인터페이스를 위한 고정된 원칙은 존재하지 않는다.

상호작용의 분석

(인간-컴퓨터(제한요소))

컴퓨터가 기능을 수행하는데 걸리는 속도!
저장과 관련된 디스크, 메모리, 네트웍 속도!
그래픽 정보의 빠른 디스플레이 속도!
네트웍 용량을 통한 자원과 파일 공유의 효율성과 속도!

상호작용의 분석

(상호작용의 유형)

적절한 활용

명령어 방식 -> 전문가 집단

메뉴방식 -> 초보자 집단

자연언어이용방식
Query Builder 사용
전문가 집단

스프레드시트
-다양한 데이터 표시

WIMP - 다양한 윈도우를 사용

포인트와 클릭 - 클릭연결

삼차원 공간 이용
- 통계자료시각화

WIMP (Windows, Icons, Menus, Pointer) or (Windows, Icons, Mice, Pull-down Menus)

사용자 인터페이스 설계 원칙 정리

설계 4대 원칙

가시성(Visiblity)의 원칙

중요 포인트, 눈에 원활한 배치, 기능의 의미 전달 쉬운 기호 사용.

대응(mapping)의 원칙

기능(버튼)의 위치와 배치가 적절해야 한다.

행동유도성(Affordance)의 원칙

체크박스, 아이콘, 버튼등의 행위의 일치

피드백의 원칙

동기부여, 지연발생시 좌절/포기 방지의 원칙

사용자 인터페이스 설계 원칙 정리

사용자 4대 원칙

기본 기준

- 버튼, 메뉴, 다이얼로그 박스의 적절한 선택과 올바른 사용과 배치
- 이용자의 주관적인 만족감, 수행속도는?
- 반응은 원활한가?

GUI구성요소와 인터페이스 평가

- 다양한 기능의 혼란감?
- 기능이 숨어있다?
- 중복메뉴의 존재?

구성요소의 배치 및 외양 인터페이스 평가

- 내가 작업하는 순서와 맞는가?
- 이용빈도는?
- 그룹구분이 쉬운가?
- 글자는 보기 편한가?

상호작용 인터페이스 평가

- 어떻게 기능을 수행하는가?
- 화면의 연결은 일관성이 있는가?
- 내가 어떤 작업을 하는가?

정보시스템의 인터페이스 모델

- 사용자의 요구는 시스템에 최대한 즉각적이고 충분한 대응을 수행
- 문자키 입력상태, 포인터의 변환, 마우스 클릭 등 모든 행위에 대하여 반응
- 명령의 실행이 지연될 경우 커서의 모양 전환 등을 통하여 적절한 정보를 표시
- 다음 예상명령이 있는 경우 사전처리를 시작해서 바로 대응
- 이용자 실수율을 최소화 (단, 속도문제를 고려하라.) – 등록하셨습니다? 메시지?
- 이용자 참여 설계(participatory design)방법의 제공

Separate of Concern을 위한 MVC 아키텍쳐 스타일 적용

- 명확히 정의된 역할에 따라 아키텍쳐 Layer 구분
  - Presentation Layer
  - Business Layer
  - Persistence Layer
- 시스템 간의 영향을 최소화 할 수 있도록 인터페이스 정의
- MVC 스타일의 적용 (화면과 기능이 완전히 분리된 형태)

프로그램 패턴화를 위한 개발 프레임웍 및 개발 표준 수립

- 공통으로 제공되는 개발 Framework과 개발자가 개발해야 할 부분이 명확히 구분되도록 함

MVC : Model-View-Controller 패턴 (디자인 패턴)

단계별 처리방안

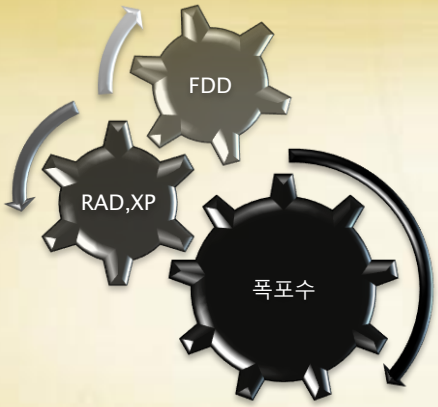
FGI : Focus group interview (소비자 면접조사), IA : Information Architecture, PIA : Perceived Information Architecture (감지된 정보 구조)

편의성에 대한 요구사항

1	가이드 없이 사용할 수 있어야 한다. 매뉴얼은 최대한 짧아서 한 자리에 읽고 충분히 이해할 수 있어야 한다.
2	사용자 인터페이스는 사용자가 수행하는 작업 순서대로 배열되어야 한다.
3	최소의 키 입력만으로 작업을 진행하여야 한다.
4	사용자가 실수해도 시스템이 죽거나 작업이 취소되거나 재부팅 되지 않아야 한다.
5	시스템은 사용자가 별도의 조치를 취하지 않아도 사용자가 한 작업을 저장해야 한다.
6	사용자는 어떤 작업이 취소되거나, 제거될 때 미리 경고를 받을 수 있어야 한다.
7	사용자 인터페이스와 메시지의 의미는 사용자가 쉽게 이해할 수 있도록 명확해야 한다.
8	사용자가 작업한 내용은 시스템이 죽어도 유지되어야 한다.
9	시스템은 네트웍 오류와 서버오류가 발생하여도 최소한의 입력작업을 수행할 수 있어야 한다.
10	시스템은 사용자가 필요한 부수적인 입력내용을 선택 입력할 수 있게 하는 기능을 최대한 갖추어야 한다.
11	사용자는 시스템의 접근시에 브라우져에 필수 도메인을 입력하는 작업 이외의 작업을 수행해서는 안된다.
12	시스템은 사용자가 마우스를 사용하지 않고 키보드만을 사용하여 데이터를 입력하거나 처리할 수 있는 방안을 제시하여야 한다.
13	시스템은 장기간 사용하는 사용자의 피로를 최소화 하기 위한 디자인적인 요소와 색채감각을 갖추고 있어야 한다.

일정의 진실

방법론 ≠ 실제개발

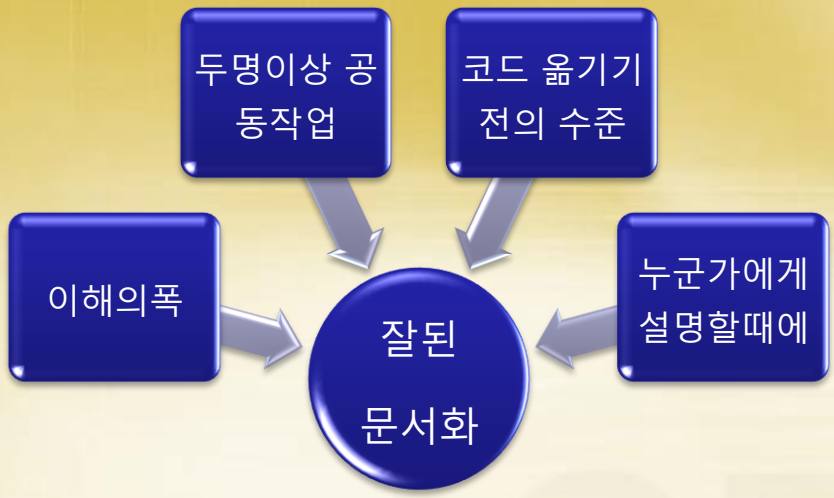


테일러 주의

테스트? 품질?

방법론의 진실

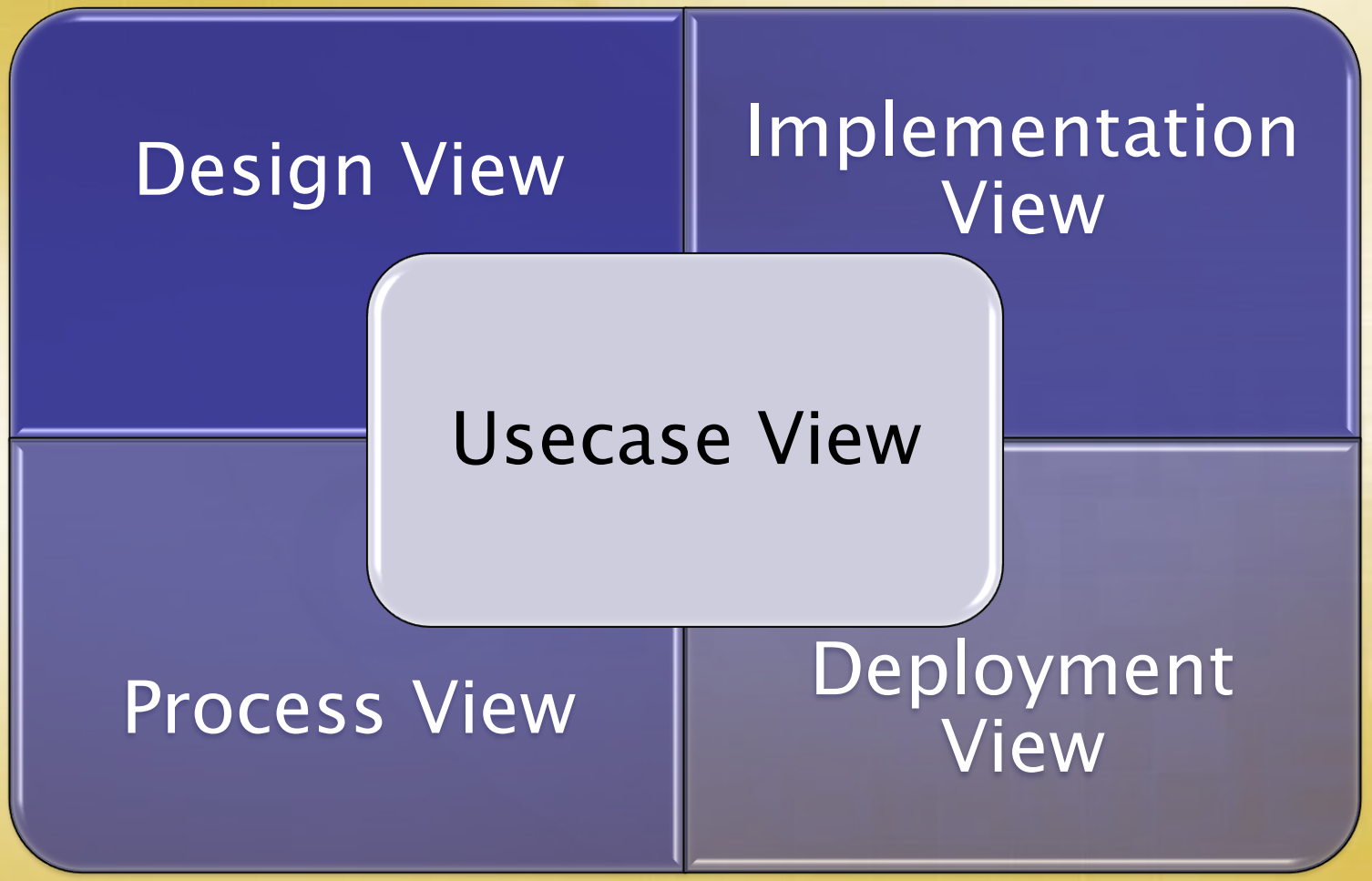
다이어그램? 문서화?


두명이상 공동작업
코드 옮기기 전의 수준
이해의폭
잘된 문서화
누군가에게 설명할때에


다그린다?
포괄적인 문서?
공정상?
잘못된 문서화
코딩에 대해?


보여주기 위한 기법
역공학툴 사용
고객만족

아키텍쳐


Design View
Implementation View
Usecase View
Process View
Deployment View

설계와 SA의 차이점…

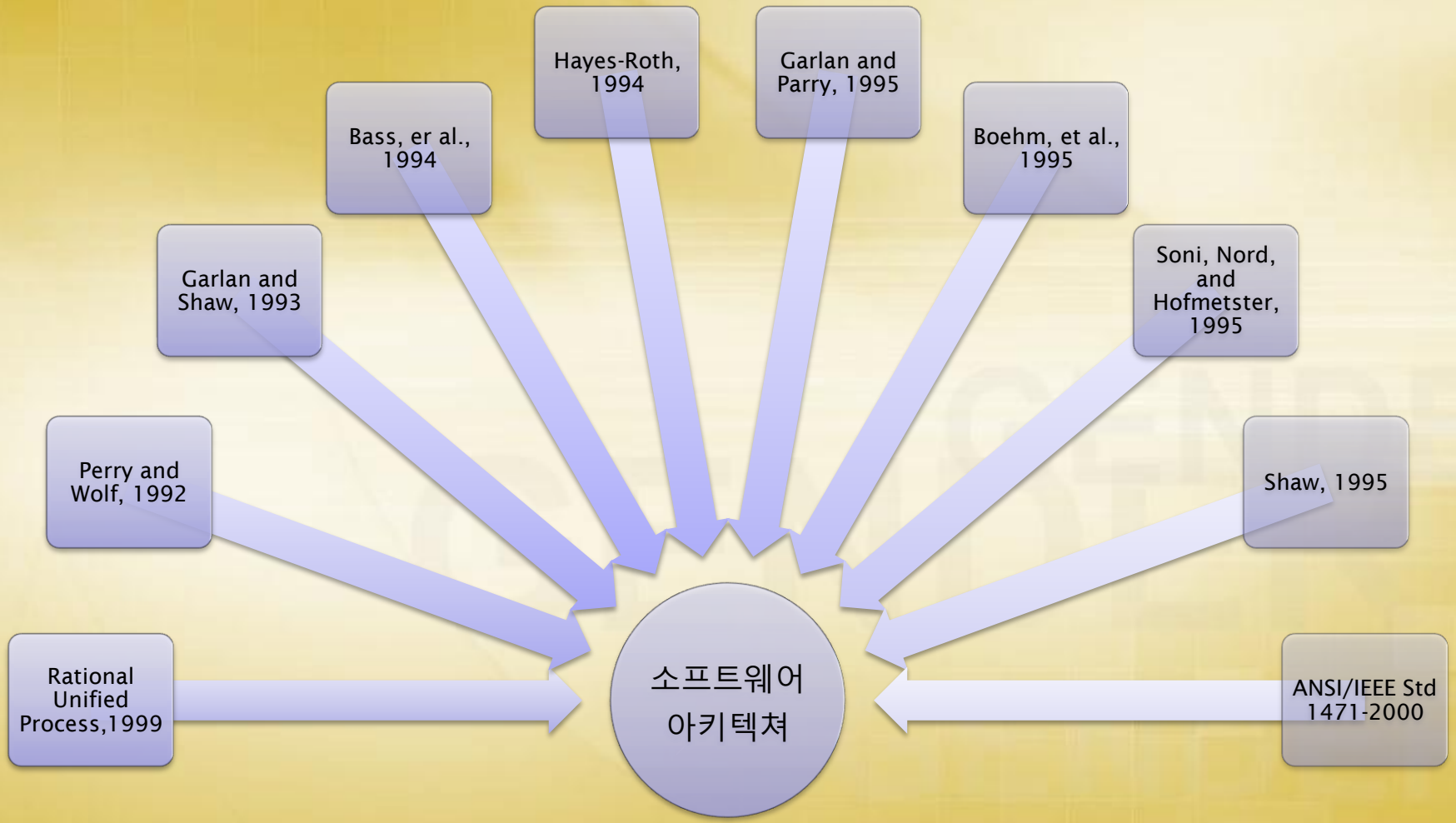
일반 개발자	아키텍트
버튼이 눌렸을때 어떻게 반응할까?	•버튼이 얼마나 자주 눌릴까? •동시에 얼마나 많은 사용자가 버튼을 누를까? •버튼을 누르는 사람들이 인트라넷 내에 있을까? 바깥에 있을까?
시스템이 이벤트에 어떻게 반응하는가?	각 이벤트간에는 어떠한 시간적인 순서와 제약적인 조건들이 존재하는가?
업무규칙이 어떠한가?	•업무규칙이 얼마나 복잡한가? •업무규칙이 얼마나 자주 변하는가? •개인에 의해 개발되는가? 독립된 팀에 의해 개발되는가? •업무 규칙들이 별개로 변하는가, 서로 영향을 주며 같이 변하는가?
데이터베이스 스키마가 어떠한가?	•스키마가 얼마나 복잡한가? •기존의 시스템에 의한 제약은 어떠한 것들이 있는가?
그 메시지에는 어떤 필드들이 있는가?	•그 메시지와 상호작용하는 외부의 시스템은 어떤 것이 있으며 그들은 어떤 특징을 가지고 있는가?

국내 프로젝트의 경향

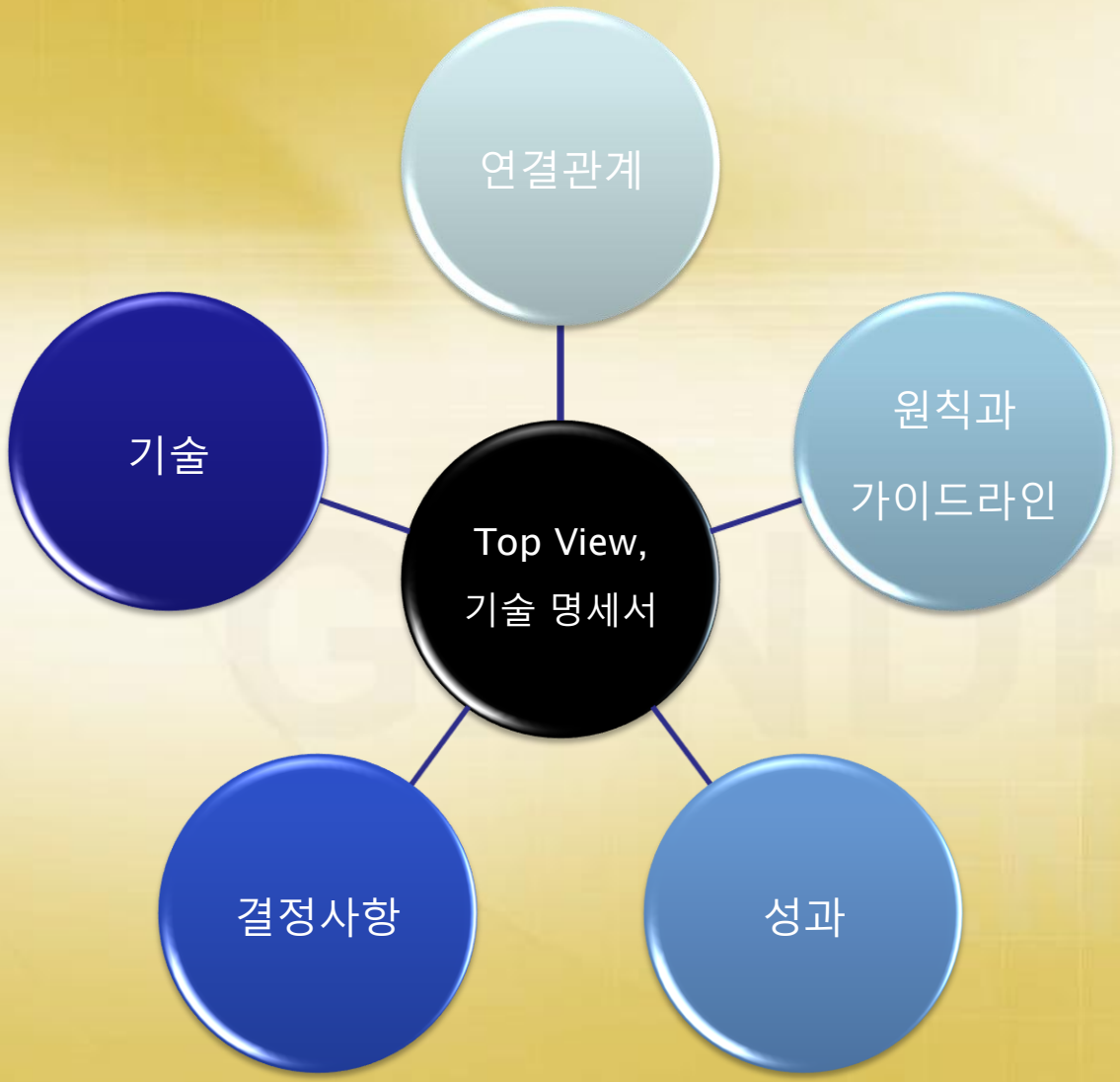

신기술 무조건 도임
프로젝트의 복잡성 관리하지 않음
계급이 깡패
CASE툴은 먼지만…
설계자가 다른일을?
요구사항관리는 하나?

소프트웨어 아키텍쳐의 목표

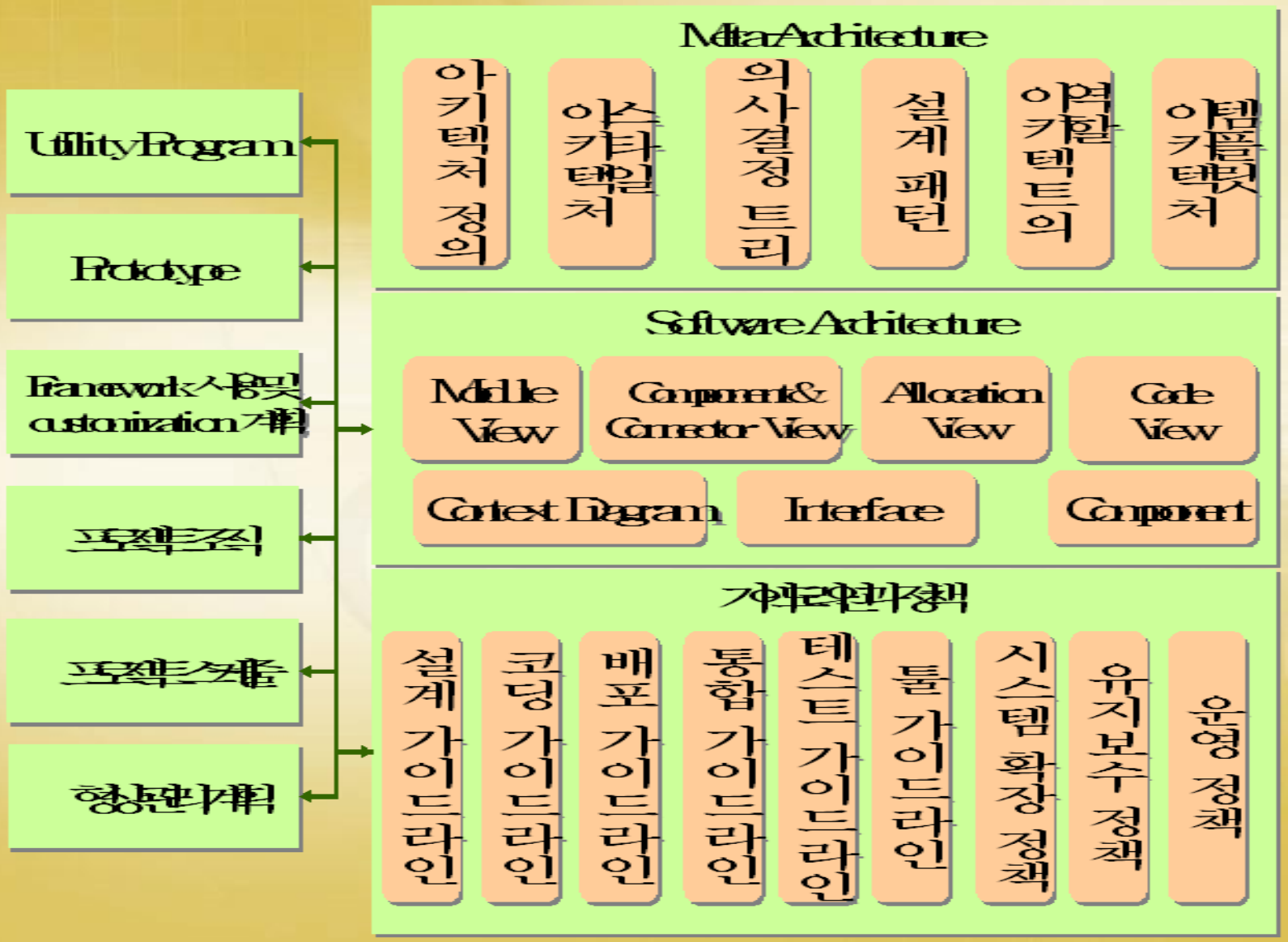
소프트웨어 아키텍쳐 정의



소프트웨어 아키텍쳐

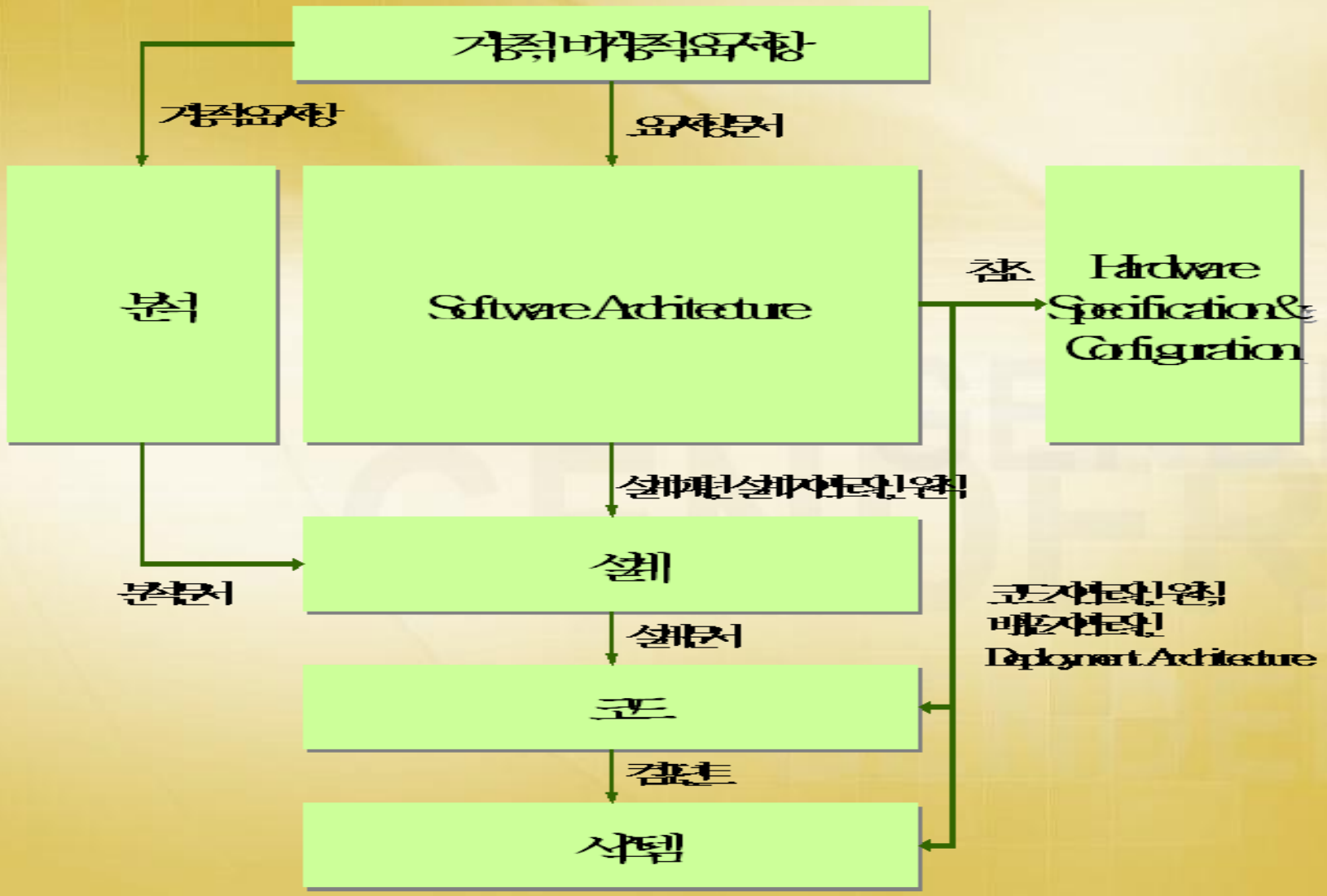


산출물

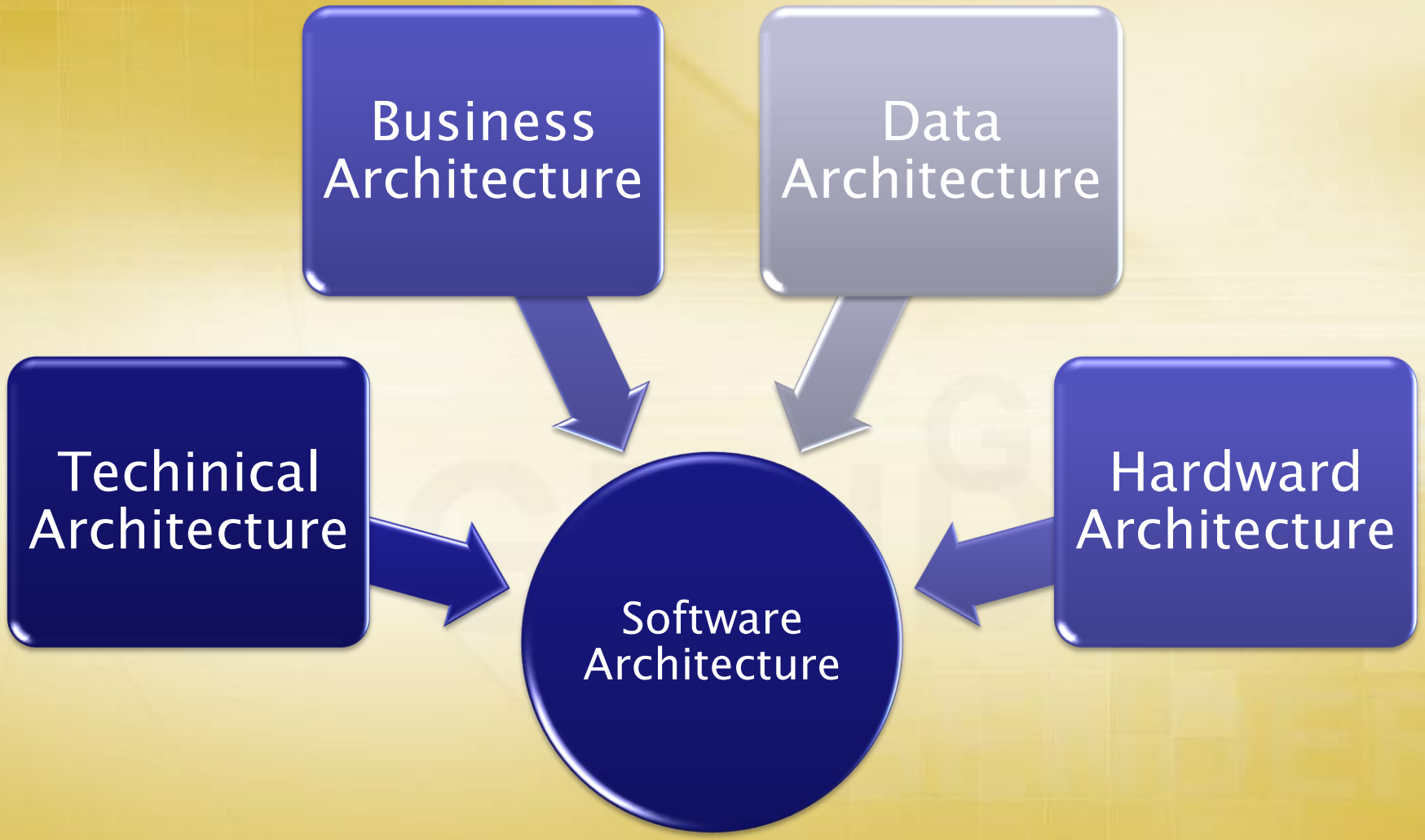

Utility Program
Prototype
Framework 사용 및 customization 계획
프로젝트 조직
프로젝트 계획
형상관리 계획
Meta Architecture
아키텍처 정의
아키타입 택일
의사결정 트리
설계 패턴
아키텍트의 역할
아키텍처 템플릿
Software Architecture
Module View
Component & Connector View
Allocation View
Code View
Context Diagram
Interface
Component
개발 관련 정책
설계 가이드라인
코딩 가이드라인
배포 가이드라인
통합 가이드라인
테스트 가이드라인
툴 가이드라인
시스템 확장 정책
유지보수 정책
운영 정책

역활

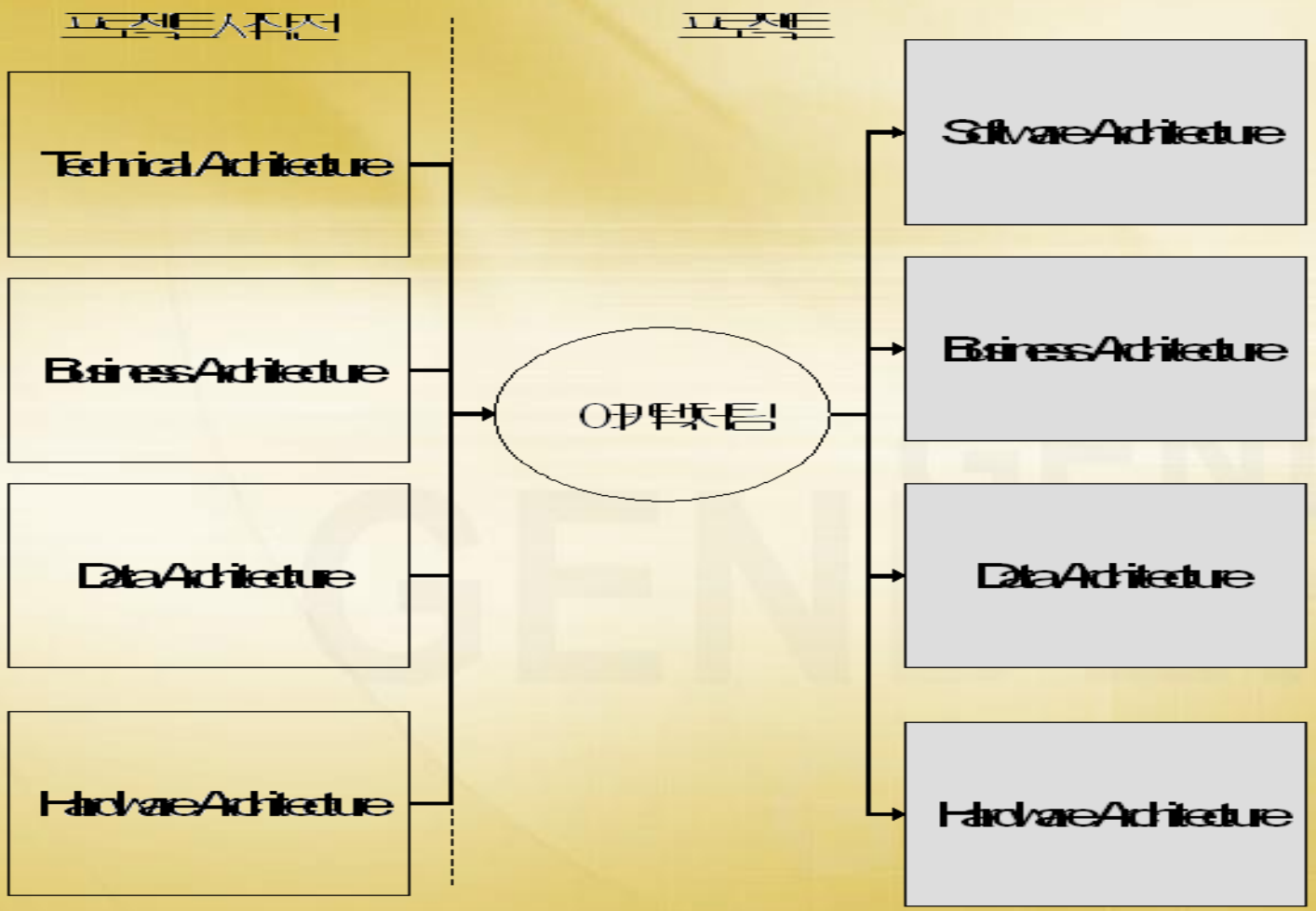
구축역활


기능적/비기능적요구사항
기능요구사항
요구분석
분석
SoftwareArchitecture
참조
Hardware
Specification&
Configuration
설계변경설계전달원칙
분석문서
설계
설계문서
코드
코드전달원칙
배포전달
DeploymentArchitecture
컴포넌트
시스템

다른 아키텍쳐와 관계

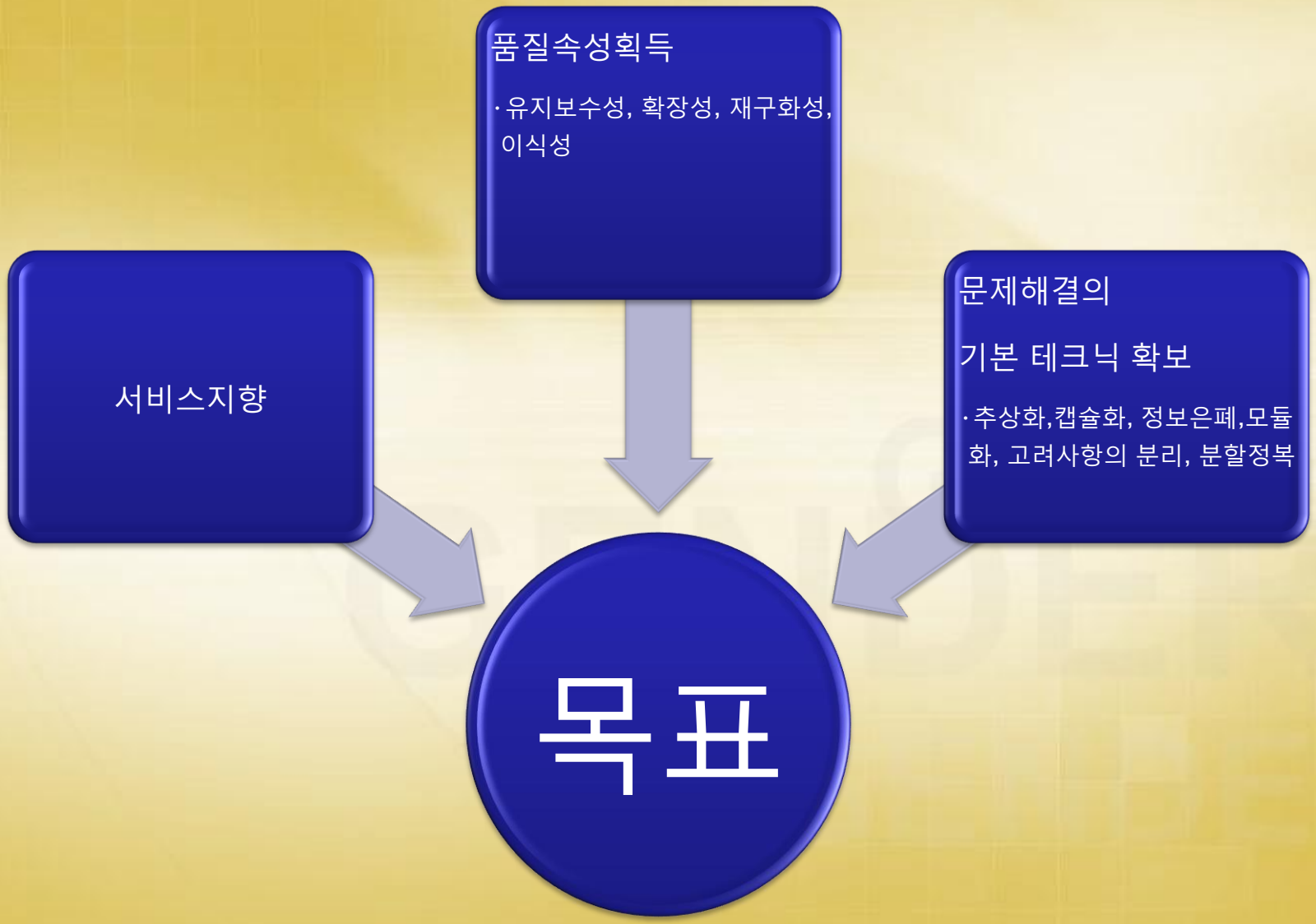

Business Architecture
Data Architecture
Techinical Architecture
Software Architecture
Hardward Architecture

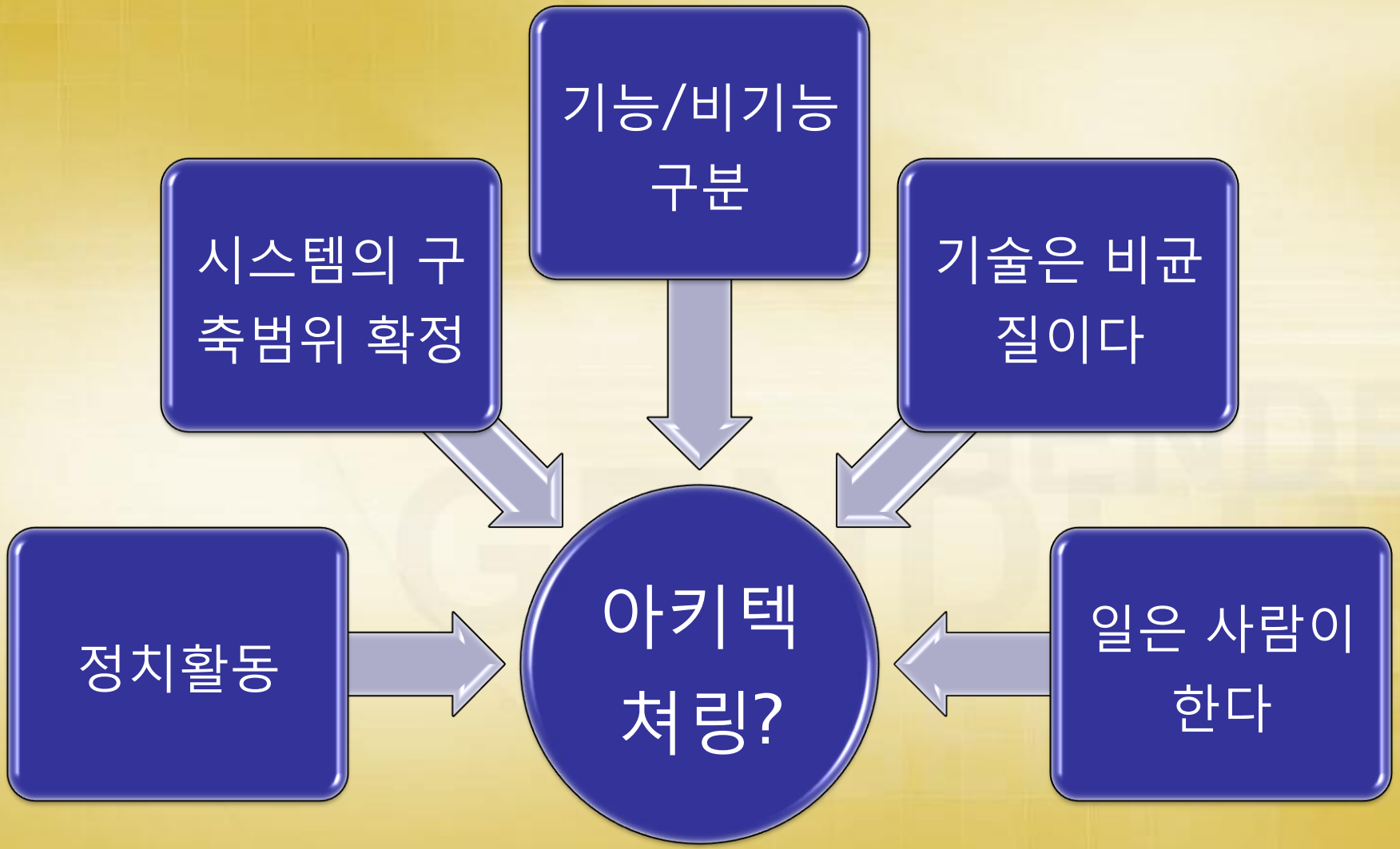
입력물/출력물


TechnicalArchitecture
BusinessArchitecture
DataArchitecture
HardwareArchitecture
SoftwareArchitecture
BusinessArchitecture
DataArchitecture
HardwareArchitecture

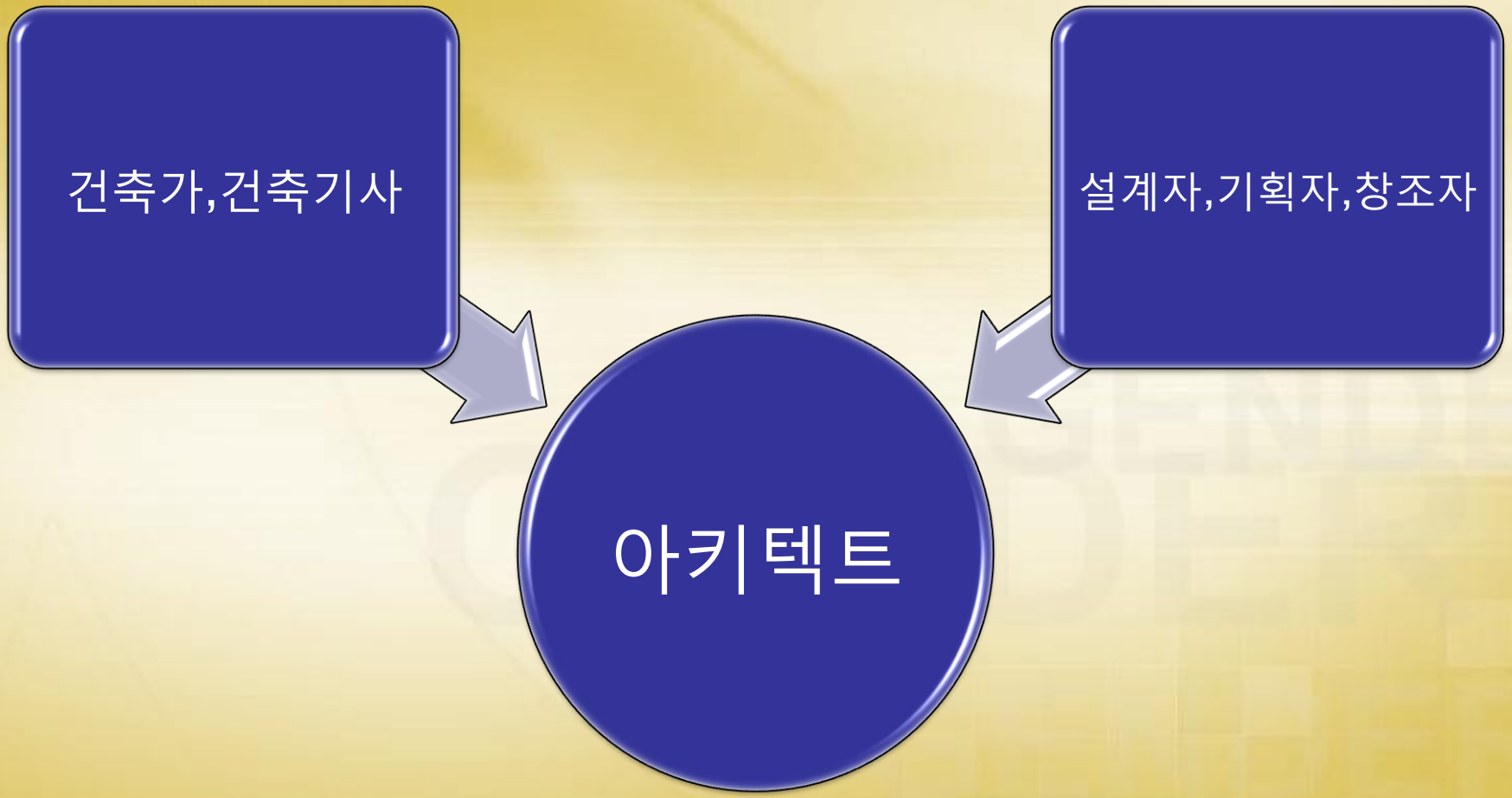
아키텍트의 역량

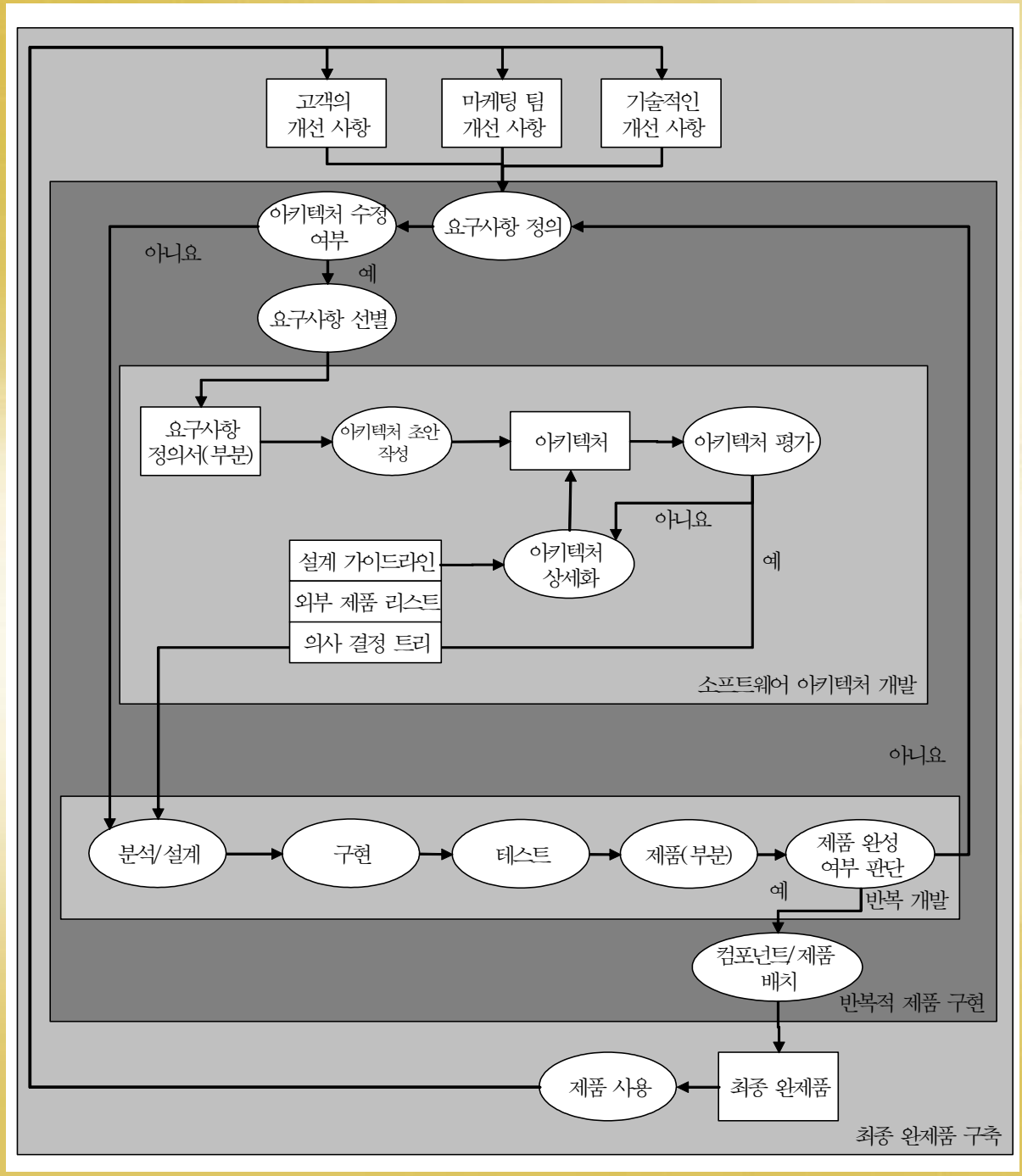
목표?


품질속성획득
·유지보수성, 확장성, 재구화성, 이식성
서비스지향
문제해결의
기본 테크닉 확보
·추상화,캡슐화, 정보은폐,모듈화, 고려사항의 분리, 분할정복
목표


기능/비기능 구분
시스템의 구축범위 확정
기술은 비균질이다
정치활동
아키텍쳐링?
일은 사람이 한다

Architect




고객의 개선 사항
마케팅 팀 개선 사항
기술적인 개선 사항
요구사항 정의
아키텍처 수정 여부
아니요
예
요구사항 선별
요구사항 정의서(부분)
아키텍처 초안 작성
아키텍처
아키텍처 평가
아니요
예
설계 가이드라인
외부 제품 리스트
의사 결정 트리
아키텍처 상세화
소프트웨어 아키텍처 개발
아니요
분석/설계
구현
테스트
제품(부분)
제품 완성 여부 판단
예
반복 개발
컴포넌트/제품 배치
반복적 제품 구현
제품 사용
최종 완제품
최종 완제품 구축

결론