

## Chapter 5

# 품질요구와 QAS

# 목 차

## Chapter 5. 품질요구

- 품질 요구 개요
- 품질요구의 명세
- Quality Attribute Scenario
- ATAM 명세

# 1. 품질 요구 개요

## 비기능 요구란?

a software requirement that **describes not what the software will do, but how the software will do it**, for example, software performance requirements, software external interface requirements, design constraints, and software quality attributes. - IEEE

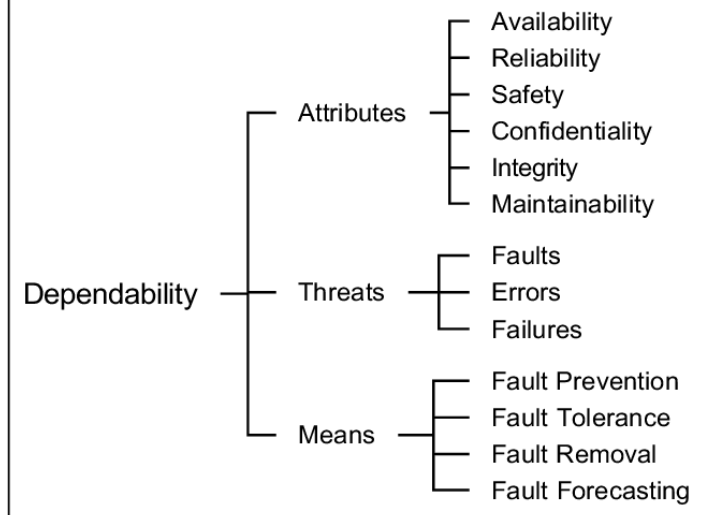
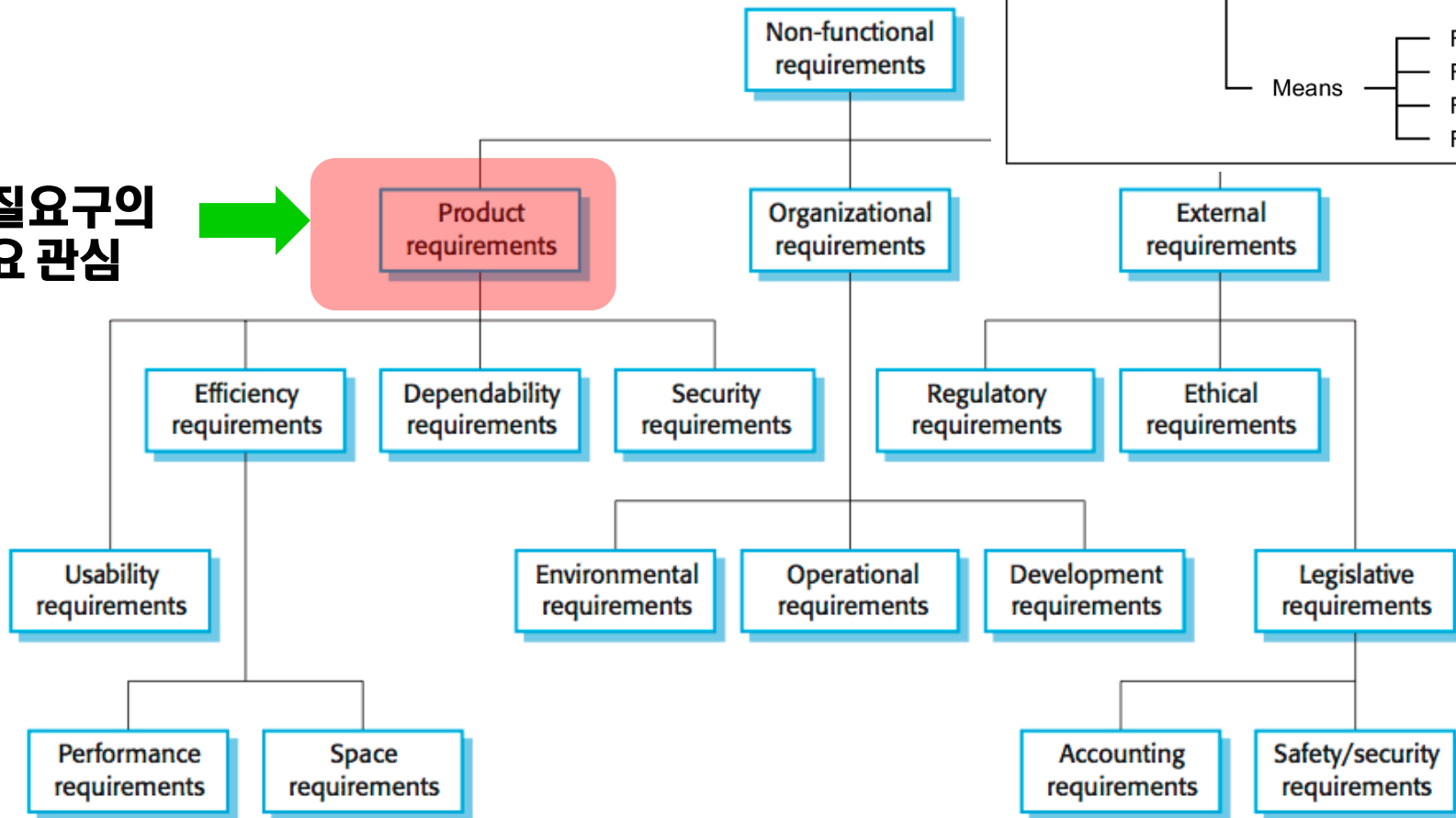
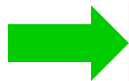
**Constraints on the services or functions** offered by the system such as timing constraints, constraints on the development process, standards, etc - **Sommerville**

- a) Describe the **non-behavioral aspects of a system**, capturing the properties and constraints under which a system must operate.
- b) The required overall attributes of the system, including portability, reliability, efficiency, testability, understandability, and modifiability.
- c) The term “non-functional requirement” is used to delineate requirements focusing on **“how good” software does something** as opposed to the functional requirements, which focus on “what” the software does. - Chung

# 1. 품질 요구 개요

## 비기능 요구의 분류(Taxonomy)

품질요구의  
주요 관심



# 1. 품질 요구 개요

## 품질요구는 비기능 요구인가?

**Quality of service requirements** are most often used to describe some of the attributes of the system or system environment. **These requirements are constraints on the solution. They are also known as non-functional requirements**

The International Institute of Business Analysis (IIBA)  
Business Analysis Body Of Knowledge (BABOK) v1.6

The important thing is not to worry about whether a requirement is a non functional or not. If you can't categorize it any other way then categorize it as non-functional.



# 1. 품질 요구 개요

## 품질을 지칭하는 용어

### □ -ilities

- understandability, usability, modifiability, interoperability, reliability, portability, maintainability, scalability, configurability, customizability, adaptability, variability, volatility, traceability, ...

### □ -ities

- security, simplicity, clarity, ubiquity, integrity, modularity,...

### □ -ness

- user-friendliness, robustness, timeliness, responsiveness, correctness, completeness, conciseness, cohesiveness, ...

### □ others

- performance, efficiency, accuracy, precision, cost, development time, low coupling, ...

# 1. 품질 요구 개요

## 품질 요구의 특징

시스템이 갖는 특성이나 행위에 대한 제약이 주요 관심임

- 시스템의 비즈니스 목적이나 특성과 관련 – SW 시스템의 성공과 관련
- Give nonfunctional requirements as much attention as functional requirements

객관적이고 측정 가능한 정량적 요구사항과 그렇지 않은 요구가 공존함

- 측정할 수 없는 것은 통제할 수 없다. (Tom DeMarco)
- 성능 및 용량 요구사항 (정량적, 객관적) vs. 사용성 (정성적, 주관적)등

고객과 품질 요구사항을 정의하는 것이 중요함



- 분석가는 고객을 위한 품질의 진정한 의미가 무엇인지 알아야 함
- SW에 대한 명확한 의미, 실현 가능성, 측정 가능한 품질 속성을 설정함
- 기술적 설계와 SW의 특징을 고려하도록 프로젝트 시작부터 품질 요구사항을 정의

# 1. 품질 요구 개요

## 품질과 기능

품질 속성은 기능, 비기능 관점을 모두 포함함

- 품질향상과 직결되는 기능요소가 있다
- 하지만, 기능만으로 품질 목표를 달성은 불가능하다



구분	Functional	Non-Functional
사용용이성	Undo, input/out data	GUI
변경용이성	Logging change history	coding style
성능	Component interface/ Load balancing	Algorithm enhancement

**Don't alter product's functionality**

- 하나 이상의 기능적 요구사항과 관련 : 상호 충돌 발생 가능성
- 기능 요구사항보다 높은 우선순위 부여 : 요구사항 변경 작업과 불일치 감소



# 1. 품질 요구 개요

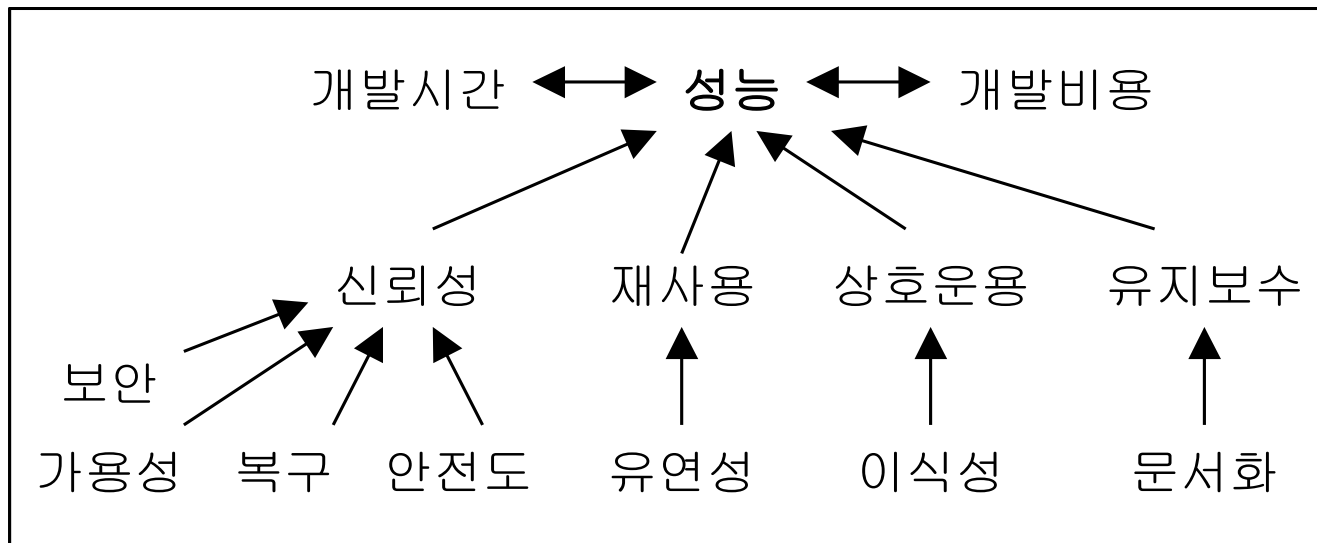
## 품질 요구의 충돌

### 품질 요구의 상호 연관

- 품질 요구를 100% 만족시킬 수는 없다. : 품질 목표를 적절한 수준에서 결정
- Positive + Negative : 예) 보안과 신뢰성
- 관련된 품질속성들은 서로 trade-off

### 연관 관계가 많은 요구사항부터 명세

- 거의 모든 품질속성은 성능에 부정적인 영향을 미친다



# 1. 품질 요구 개요

## 품질 요구의 예

- 시스템은 주중 오전 6시에서 자정까지는 99.5%, 그 이외의 시간은 최소 99.5%까지 사용할 수 있어야 한다
- 응용 프로그램을 위한 프로세서 용량과 RAM 중에서 20%는 최대 부하시점에서도 사용되지 않아야 한다
- 감사접속 권한을 가진 자만이 고객 거래자료를 볼 수 있다
- 사용자가 파일을 저장하기 전 편집기에 에러가 발생하면 편집 중이던 모든 변경내용을 에러발생 5분 전까지로 복구한다
- 메뉴 파일의 모든 기능은 Ctrl+다른키를 사용하는 단축키가 정의되어야 한다
- 함수 호출은 3 단계 이상 중첩되지 않는다
- 모듈의 Cyclomatic Complexity는 20을 넘지 않는다
- 온도관리 주기는 0.8초 이내에서 수행한다
- 모든 웹 페이지는 10Mbps LAN 접속에서 5초 이내로 다운로드 한다

# 1. 품질 요구 개요

## 품질 요구의 예 : HMC-PMS

### ■ MHC-PMS ( Mental Health Care Patient Management System )

#### Product requirement

The MHC-PMS shall be available to all clinics during normal working hours (Mon–Fri, 0830–17.30). Downtime within normal working hours shall not exceed **five seconds in any one day.** → Availability

#### Organizational requirement

Users of the MHC-PMS system shall authenticate themselves using their health authority identity card. → Security

#### External requirement

The system shall implement patient privacy provisions as set out in HStan-03-2006-priv. → Standard

# 1. 품질 요구 개요

## 품질 요구의 예 : HMC-PMS

### Product requirement

The system **should be easy to use by medical staff** and should be organized in such a way that user errors are minimized. → Usability

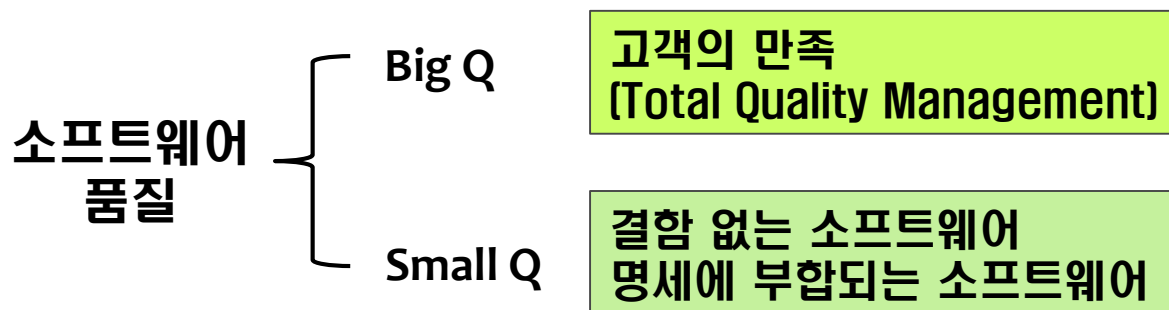
Medical staff shall be able to use all the system functions after **four hours** of training. After this training, the average number of errors made by experienced users **shall not exceed two per hour of system use**. (Testable non-functional requirement) → How much usable?

# . 품질 요구 개요

## 품질속성 (Quality Attributes)

### ❑ 소프트웨어 품질의 특성

- 소프트웨어 품질은 복합적인 요소가 포함된 다중 개념임
- 단순히 오류가 없다고 해서 품질 좋은 소프트웨어라 말할 수 없음



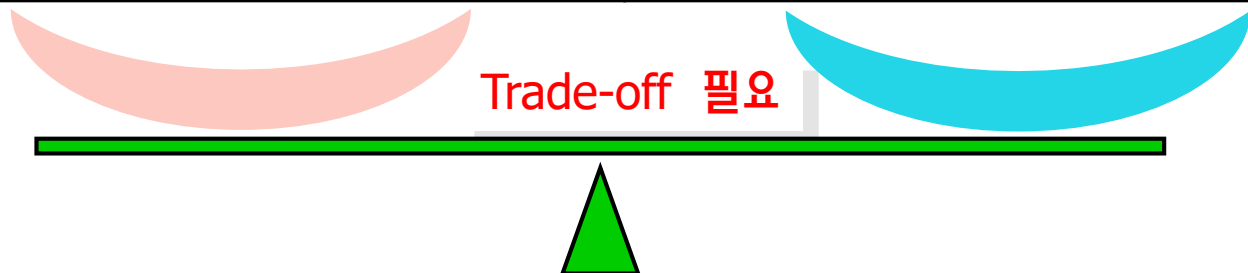
### ❑ 품질속성 (Quality Attribute)

- 소프트웨어는 여러 측면을 나타내는 속성 (예: LOC, 복잡성, 성능, 메모리 사용량 등) 이 있으며 이 중 어떤 것은 품질과 연관이 있음
- 품질속성은 소프트웨어의 품질 관련 측면을 나타내고 특징짓는 속성을 말함

## 2. 품질 요구 카테고리

### 품질속성의 구분 (계속)

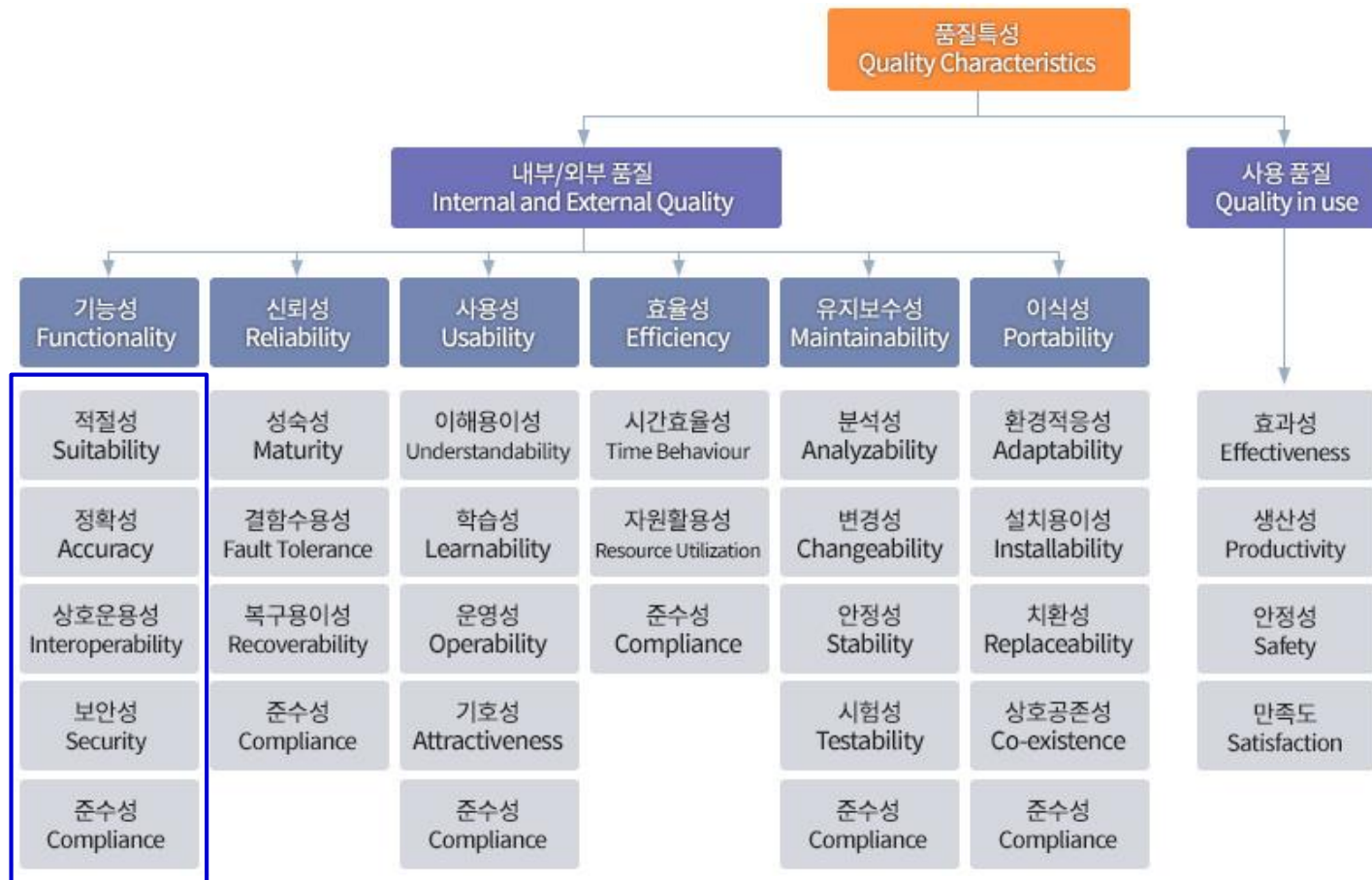
	외부 품질	내부 품질
대 상	<ul style="list-style-type: none"> <li>• 소비자/고객 관점</li> </ul>	<ul style="list-style-type: none"> <li>• 개발자, 유지보수자</li> </ul>
품질 특성	<b>Run-time 품질</b> <ul style="list-style-type: none"> <li>• 목적 만족을 위한 성질, 성능</li> <li>• 시스템 행위에 대한 사용자 평가</li> </ul>	<b>Development-time 품질</b> <ul style="list-style-type: none"> <li>• 개발 조직의 산출물 속성 만족</li> <li>• 프로세스 산출물에 대한 내부 평가</li> </ul>
품질 요소	Usability, Efficiency, Safety, Reliability, Availability, Performance 등	Modifiability, Extensibility, Composability, Reusability, Testability 등
품질 Source	<ul style="list-style-type: none"> <li>• User Objectives, Values, Concerns</li> <li>• Competitive Analysis of Features</li> </ul>	<ul style="list-style-type: none"> <li>• 개발 조직의 Objectives, Values, Concerns, 제약사항</li> <li>• 경쟁업체 및 산업 동향</li> </ul>



## 2. 품질 요구 카테고리

### ISO/IEC 9126

- ❑ 소프트웨어 품질의 특성을 정의하고 품질 평가의 Metrics를 정의한 국제 표준
- ❑ 사용자, 평가자, 개발자 모두에게 소프트웨어 제품의 품질을 평가하기 위한 지침



## 2. 품질 요구 카테고리

### ISO/IEC 25010

- ❑ 기존 ISO/IEC 9126에서 ISO/IEC 25010으로 개정됨
- ❑ 주특성이 기존 6개에서 8개로 증가됨
  - 기능 적합성, 신뢰성, 사용성, 유지보수성, 이식성, 실행 효율성, **호환성**, **보안성**





## 2. 품질 요구 카테고리

### ISO/IEC 25010 기능성 부특성

- ❑ 기능성 (functionality) : 소프트웨어를 포함하고 있는 시스템의 기능적 행위와 같은 속성을 측정함
- ❑ 기능을 사용할 때 오류가 있는가?, 기능들이 모두 잘 동작하는가?
- ❑ 기능성의 부 특성

기능성숙도 (completeness)	명시된 요구사항이 충실히 구현되었는가?
기능정확성 (Correctness)	사용자가 기대한 결과와 실제 결과는 얼마나 정확한가?
기능타당성 (Appropriateness)	사용자 목적달성에 얼마나 도움을 주고 있나?

## 2. 품질 요구 카테고리

### Availability (가용성)

- ❑ 서비스가 이용가능하고 완벽하게 작동하는 기간에 대한 비율 [예: 가용성 99.9%]을 말하며, 아래의 의미를 포함함
  - 서비스가 중단되지 않고 운영될 수 있는 능력
  - 사용이 필요할 때 시스템이 지정한 기능을 수행할 수 있는 확률
  - 총 운영 시간에 대한 시스템 가동시간의 비율
- ❑  $Availability (\%) = MTBF / (MTBF + MTTR) \times 100$ 
  - MTBF: Mean Time Between Failure (평균고장발생간격),
  - MTTR: Mean Time To Repair (평균 복구시간)
- ❑ 가용성 요구는 웹사이트나 클라우드 기반 응용, 시간대가 흩어져 제공되는 애플리케이션에서 특히 중요함
- ❑ 예: 시스템은 오전 5시와 자정 사이에 95%이상, 오후 3시~5시 사이에는 99%이상 이용할 수 있어야 한다
- ❑ 가용성은 가끔 SLA (Service Level Agreement)로 규정되기도 함

## 2. 품질 요구 카테고리

### Availability (가용성) – 계속

#### □ 가용성 도출과 관련한 질문

- 가용성을 제공하기 위해 시스템에서 가장 중요한 부분은?
- 사용자가 시스템을 사용하지 못함으로써 발생하는 비즈니스의 결과는?
- 정기적인 유지보수 예약을 해야 하는 경우는 언제인가?
- 유지보수 시간의 최소 및 최대 기간은 언제인가?
- 유지보수 기간 동안 사용자 접근시도는 어떻게 관리되나?
- 시스템을 사용할 수 없을 경우 사용자에게 필요한 알림은 무엇인가?
- 기능 그룹에서 가용성과 관련된 기능은 무엇인가?

## 2. 품질 요구 카테고리

### Integrity (무결성)

- ❑ 정보손실을 방지하고 시스템에 입력된 데이터의 정확성을 지키는 문제를 다룸
- ❑ 데이터 무결성은 데이터의 정확도 및 형식을 보장하기도 함
- ❑ 날짜 필드형식, 올바른 데이터 유형이나 길이, 유효값의 확인, 어떤 필드의 특정한 값 등
- ❑ 파일 백업을 수행한 후 복사본과 원본간 비교를 통해 불일치가 발생시 보고함
- ❑ 시스템의 인가되지 않은 데이터의 추가, 삭제, 변경을 방지해야 함
- ❑ 무결성 도출과 관련한 질문
  - 데이터의 변경이 완벽히 이루어지거나 전혀 이루어지지 않도록 보장하나?
  - 데이터 변경내용의 지속성을 보장하는가?
  - 데이터의 백업을 수행한다 (어떤 주기로? 자동 아니면 필요에 따라? 등)
  - 데이터 복원: 어떤 데이터를 얼마 동안 보관하며 삭제 조건은 무엇인가?

## 2. 품질 요구 카테고리

### Interoperability (상호운영성)

- ❑ 시스템이 다른 장비와, 소프트웨어와 얼마나 쉽게 데이터와 서비스를 교환할 수 있는지, 외부 하드웨어와 손쉽게 통합할 수 있는지를 말함
- ❑ 예: 화학약품 관리 시스템은 ChemDraw 3.0버전과 MarvinSketch 5.0 버전 이상의 도구에서 그린 유효한 화학약품 구조를 추가할 수 있어야 한다
- ❑ 상호운영성은 외부시스템 인터페이스 요구사항으로 표현할 수 있다
- ❑ 상호운영성 도출과 관련한 질문들
  - 이 인터페이스는 어떤 시스템과 연계되는가?
  - 다른 시스템과 데이터를 교환하는데 필요한 표준 데이터 형식은 무엇인가?
  - 시스템과 연결되는 하드웨어 구성요소는 무엇인가?
  - 다른 시스템이나 장비로부터 전달받거나 처리되는 메시지는 무엇인가?
  - 상호운영성을 가능하도록 하는데 필요한 표준 통신 규약은 무엇인가?

## 2. 품질 요구 카테고리

### Performance (성능)

- 성능은 사용자의 조회 및 동작에 대한 시스템의 민첩한 정도를 말하며 아래 표의 내용을 포함한다.
- 성능은 프로그램의 실행시간에 발견할 수 있으므로 외부 품질 속성이다.

성능차원	예
응답시간	웹 페이지를 표시하는데 걸리는 시간 (초)
처리량	초당 신용카드 거래량
데이터 용량	데이터베이스에 저장된 레코드의 최대 수
동적 용량	소셜미디어 웹사이트의 최대 동시 접속자수
대기시간	음악녹음 및 제작 소프트웨어의 지연시간
성능저하모드 또는 과부하 조건에서의 동작	비상 전화시스템에 엄청난 양의 통화를 유발하는 자연 재해

## 2. 품질 요구 카테고리

### Reliability (신뢰성)

Safety Integrity Level	Risk Reduction Factor	Probability of Failure on Demand
SIL 4	100,000 to 10,000	$10^{-5}$ to $10^{-4}$
SIL 3	10,000 to 1,000	$10^{-4}$ to $10^{-3}$
SIL 2	1,000 to 100	$10^{-3}$ to $10^{-2}$
SIL 1	100 to 10	$10^{-2}$ to $10^{-1}$

- ❑ 소프트웨어가 실행되는 특정시간 동안 장애가 발생하지 않는 것
- ❑ 따라서 어떤 기준 시점에서부터 서비스 수행의 지속성에 대한 척도로 장애가 발생하기까지의 시간, 즉 MTTF(Mean Time To Failure)로 표현
- ❑ 신뢰성 문제는 잘못된 입력이나 소프트웨어 코드 오류, 필요상황에 이용할 수 없는 구성요소, 하드웨어 장애로 발행할 수 있음
- ❑ 예: 소프트웨어 장애로 인한 실험 누락은 1,000번의 실험 중 5를 넘을 수 없다.
- ❑ 신뢰성은 다른 부류와 같이 후행성이기 때문에 시스템 운영에 잠시 문제가 없다 하더라도 이를 달성했다고 말할 수 없다
- ❑ 예: 카드 판독기의 평균 무고장 시간은 최소 90일 이어야 한다
- ❑ 신뢰성 도출에 관련한 질문
  - 시스템이 충분한 신뢰성을 갖는지 여부를 어떻게 판단하나?
  - 시스템이 특정작업을 수행도중 장애가 발생할 때 이에 대한 결과는 무엇인가?
  - 어떤 조건에서 장애가 비즈니스 운영에 심각한 영향을 주는가?
  - 시스템이 비즈니스 운영에 영향을 미치기까지 얼마나 정지될 수 있는가?

## 2. 품질 요구 카테고리

### Usability (사용성)

- 사용성은 시스템의 입력을 준비시키고, 동작시키며, 출력하는데 드는 노력을 측정함
- 소프트웨어 사용성은 사용용이성 외에 학습용이성, 기억용이성, 오류 방지/처리/복원성, 상호작용 효율성, 접근성, 인체공학 등 방대한 주제를 다룸
- 동일 범주 내에서 충돌 요소가 존재함 (예: 학습용이성과 손쉬운 사용성, 초보자 vs 숙련 사용자의 경우)

학습용이성	손쉬운 사용
자세한 안내	키보드 단축키
마법사	풍부한 사용자 정의 및 메뉴
눈에 띄는 메뉴옵션	항목 자동완성, 오타 자동수정
도움말 화면 및 툴팁	매크로 기능
다른 시스템과의 유사성	폼 필드 자동채우기
제한된 수의 옵션과 위젯	명령줄 인터페이스



## 2. 품질 요구 카테고리

### Usability (사용성) – 계속

#### ■ 사용성의 예

- 숙련된 사용자는 카탈로그의 화학약품을 3분내 요청할 수 있다
- 파일 메뉴의 모든 기능은 컨트롤 키와 다른 키 조합으로 이루어진 단축키를 갖는다
- 화학약품 관리 시스템을 사용한 적이 없는 화학자의 95%는 15분 이내의 간단한 교육을 통해 화학약품을 제대로 요청할 수 있다

#### ■ 사용성 지표의 예

- 특정 사용자가 어떤 태스크를 완료하는데 필요한 평균시간
- 사용자가 별도의 도움 없이 온전히 완료할 수 있는 트랜잭션 수
- 태스크 완료시 사용자가 발생시킬 수 있는 에러의 양
- 특정 기능을 찾는데 필요한 시도 수
- 작업수행에 필요한 지연시간 혹은 대기시간
- 특정 정보를 얻는데 필요한 상호작용 (마우스 클릭, 키입력, 터치 화면 등) 수

## 2. 품질 요구 카테고리

### Efficiency (효율성)

- ❑ 시스템이 프로세서 자원이나 디스크 용량, 메모리, 통신 대역폭 등을 얼마나 잘 이용하는지에 관한 척도임
- ❑ 효율성은 외부 품질 속성과 밀접한 관련이 있음
- ❑ 효율성과 성능은 계산이나 기능을 다른 시스템 구성요소로 분산시키는 것을 결정하는데 영향을 미친다
- ❑ 효율성은 다른 품질 요소를 달성하는데 부정적인 영향을 미친다
- ❑ 예: 계획된 최대 부하조건에서 애플리케이션은 최소 30%의 프로세서 자원과 메모리를 이용할 수 있어야한다
- ❑ 효율성에 관련된 질문
  - 현재 최대 동시사용자는 어떻게 되며, 향후 예상 수치는?
  - 응답시간이나 다른 성능지표가 얼마나 감소하면 비즈니스에 불이익을 주는가?
  - 정상 또는 극한의 동작조건에서 시스템이 얼마나 많은 명령을 동시에 수행할 수 있어야 하나?

## 2. 품질 요구 카테고리

### Modifiability (수정용이성)

- ❑ SW 설계나 코드가 얼마나 쉽고, 수정하기 쉬우며, 확장이 용이한가를 다룸
- ❑ 수정용이성은 소프트웨어의 유지보수와 직접 연결되는 품질 속성임
- ❑ 예: 유지보수자는 화학약품 보고 규정 준수를 위해 10시간 이하의 개발을 통해 기존 보고서를 수정할 수 있어야 한다
- ❑ 예: 함수 호출은 두 단계 이상 중첩될 수 없다

유지보수 유형	설명
수정	결함수정
확장성	신규 비즈니스 요구사항을 만족하도록 기능 향상 및 수정
적응성	새로운 기능 추가 없이 변경된 운영환경에서 작동하도록 시스템을 수정
현장지원	해당 운영환경에서 고장 수리 및 장치 정비, 수리

## 2. 품질 요구 카테고리

### Scalability (확장성)

- 성능이나 정확성 차이 없이 더 많은 사용자, 데이터, 서버, 트랜잭션, 네트워크 트래픽 등 다른 서비스를 수용할 수 있는 애플리케이션의 능력을 나타냄
- 확장성은 하드웨어, 소프트웨어 모두에 영향을 준다
- 하드웨어의 경우 더 빠른 컴퓨터의 도입, 메모리나 디스크 공간 추가, 서버추가, DB 미러링 등을 포함
- 소프트웨어의 경우 데이터 압축, 알고리즘 최적화, 기타 성능 튜닝 등을 포함
- 예: 긴급 전화 시스템의 수용능력은 12시간 이내 하루 500통에서 2,500통으로 증가시킬 수 있어야 한다
- 예: 웹사이트는 최소 2년 이상 분기당 30%의 뷰 증가를 처리할 수 있어야 하며, 사용자는 성능 저하를 인지하지 못해야 한다

### 3. 품질 요구의 추출

#### Quality measure/metric

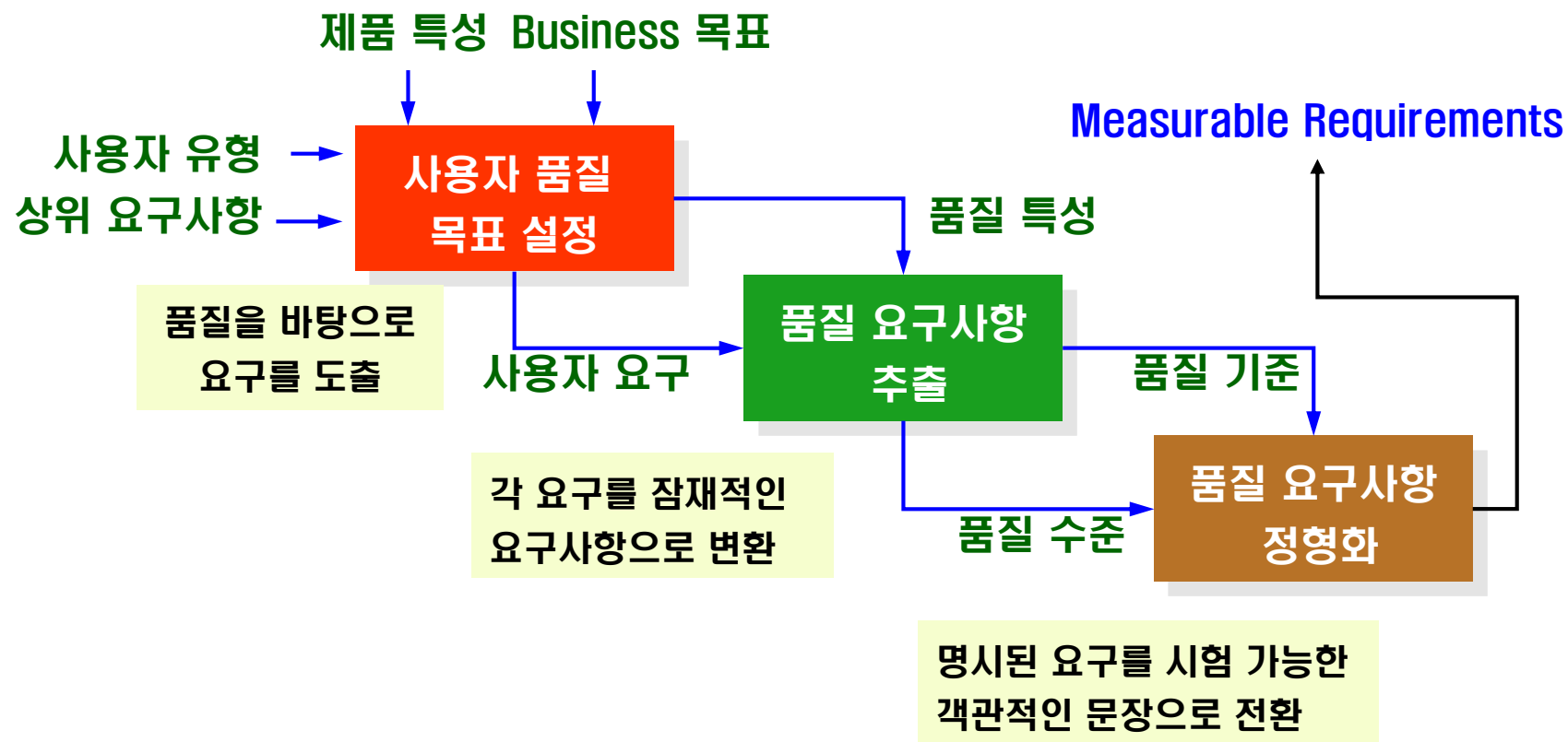
##### ▣ Measure vs. Metric

- 비기능 요구(NFR)는 개발된 시스템의 품질을 정량적으로 규정함
- 일반적으로 measure와 metric은 혼용해서 사용되는 개념임
- 경우에 따라 measure는 구체적이고 객관적인 품질요소를, metric은 주관적이며 추상적인 요소를 표현함 (NIST)
- measurement는 quality value를 실제로 얻는 과정을 말함

Property	Measures
Ease of use	- Training time, Number of help frames
Reliability	- Mean time to failure, - Rate of failure occurrence
Robustness	- Time to restart after failure - Percentage of events causing failure
Portability	- Percentage of target dependent statements - Number of target systems

### 3. 품질 요구의 추출

#### SW 품질 요구 추출의 절차



### 3. 품질 요구의 추출

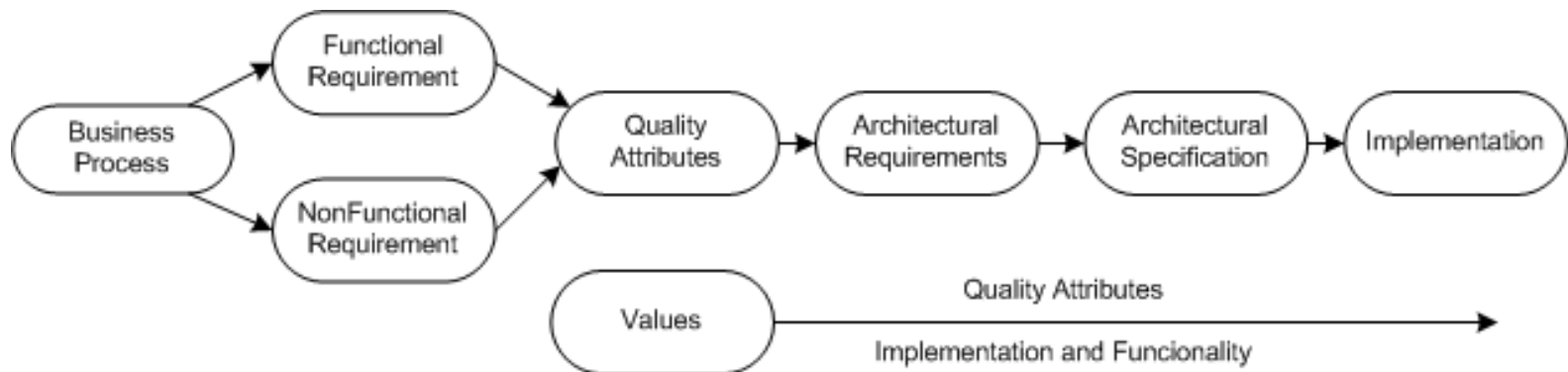
## Architecturally Significant Requirement (ASR)

#### □ Architecturally Significant Requirement

- a subset of requirements that affects the architecture of a system  
in measurably identifiable ways (Wiki)
- 아키텍처에 심각한 영향을 미치는 요구를 지칭함
- ARS는 종종 시스템이 제공하는 성능, 보안성, 변경용이성, 가용성, 사용 편의성과 같은 품질요구의 형태를 가짐
- Architect는 ASR을 식별할 책임을 가짐
- ASR에 관한 가장 중요한 정보는 이해당사로부터 나옴
  - ◆ 후보 ASR을 찾는 가장 명백한 위치 : 인터뷰 문서, 사용자 스토리, 유스 케이스

### 3. 품질 요구의 추출

## Architecturally Significant Requirement (ASR)



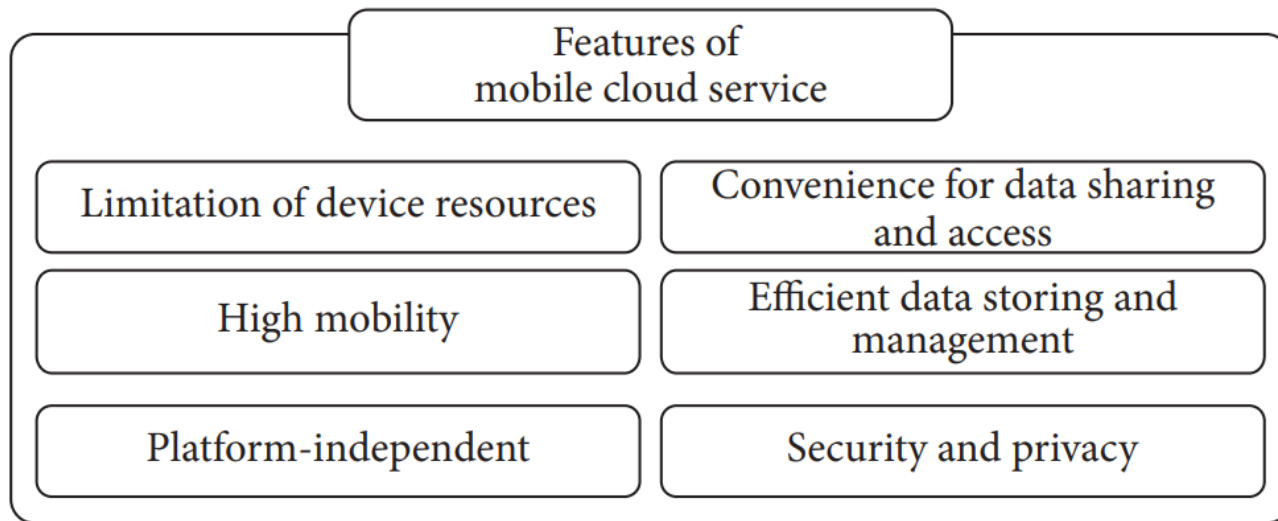
Use Case	Quality Attribute Scenario		
Description	Scenario Refinement	Stimulus	
References		Stimulus Source	
Actors		Environment	
Prerequisites		Artifact	
(Dependencies) & Assumptions		Response	
Steps		Response Measure	
Variations (optional)			
Quality Attributes			
Issues			



### 3. 품질 요구의 추출

#### 제품 품질속성의 식별 - 모바일 클라우드 서비스의 예

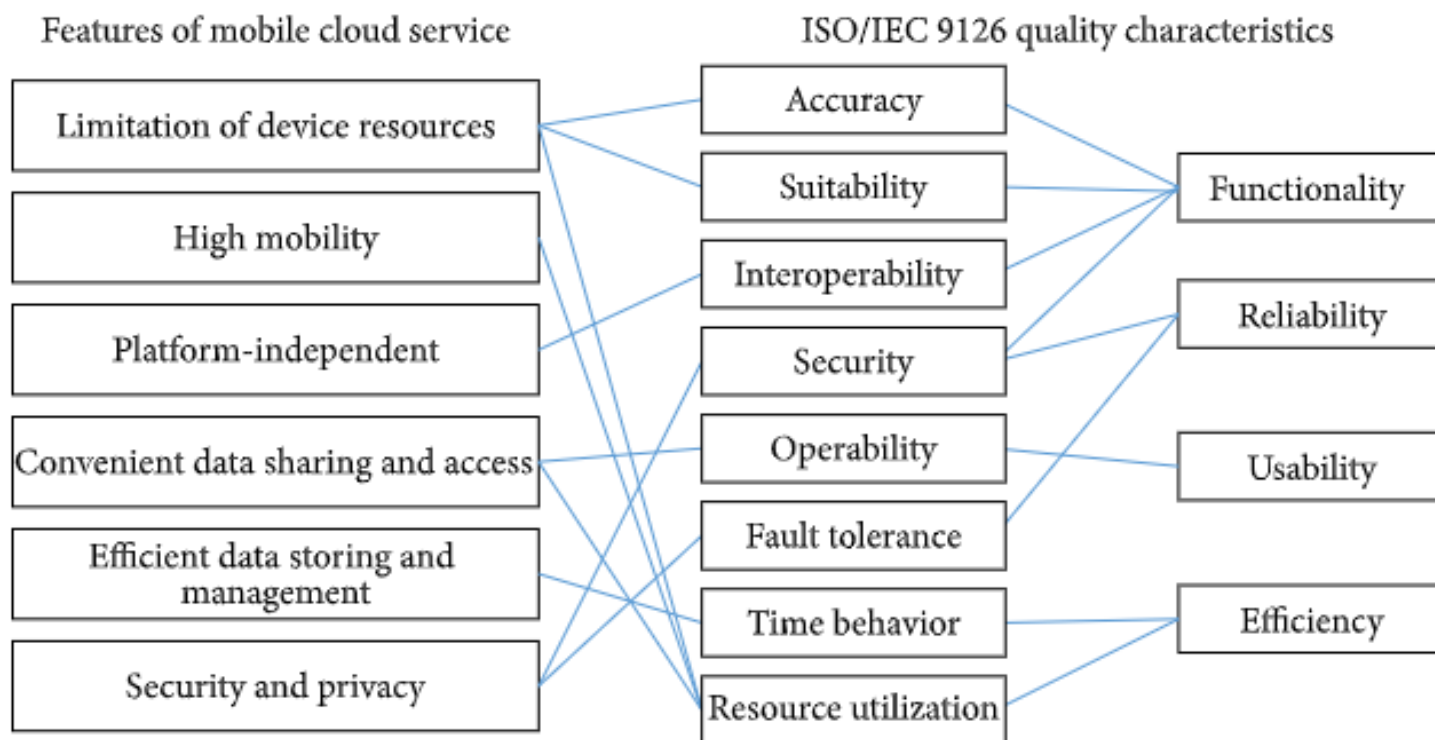
- ❑ 모바일 클라우드 서비스는 사용자에게 웹 기반의 응용서비스를 제공
- ❑ 이 경우 ISO 9126의 품질속성 매핑을 통해 제품품질의 측정 및 모니터링이 가능함
- ❑ Step1 : 모바일 클라우드 서비스의 required assurance 요소 식별



### 3. 품질 요구의 추출

#### 제품 품질속성의 식별 - 모바일 클라우드 서비스의 예

- Step 2: 모바일 클라우드 서비스의 ISO 9126의 품질속성 매핑을 통해 제품품질의 측정을 위한 요소 정의 및 메트릭 정의



### 3. 품질 요구의 추출

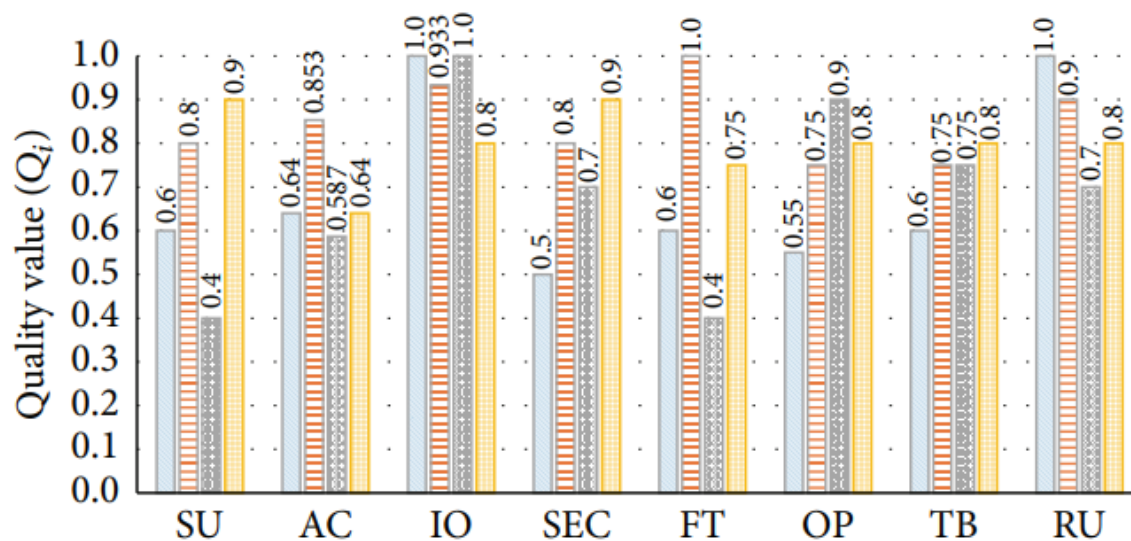
#### 제품 품질속성의 식별 - 모바일 클라우드 서비스의 예

- Step3 : 서비스에 대한 시나리오별 required assurance 요소 측정을 위한 메트릭 측정

$$\text{Suitability } SU = 1 - \frac{\text{number of missing functions}}{\text{number of required functions}}$$

$$\text{Accuracy } AC = 1 - \frac{\text{number of exceed expectations}}{\text{number of attempts for required functions}}$$

$$\text{Interoperability } IO = 1 - \frac{\text{no. of failures when data exchanges}}{\text{no. of total data exchanges}}$$



# 4. 품질 요구 명세

## 품질 요구사항의 정형화

최적의 명세를 위한 규칙이 없다

- 정량화가 어렵고 비정형적
- 고객이 무리한 수준의 품질완성도 요구를 경계함 (ex : error-free system)
- 비전형적 기준은 낮은 품질로 인한 요구사항 추가/변경 거부 명분을 약화시킴

소프트웨어 품질인증의 기준선 제공을 위한 요구사항의 정형화

- 시스템의 품질 평가 목표에 대한 사용자와 Tester 간의 동의 획득
- 시험자가 시스템의 요구사항에 대한 충분한 이해 확보

SMART 요구사항 (Mannion & Keepence, 1995)

- **Specific**      명확한 표현, 일관된 용어, 간결한 표현, 적절한 상세 표현
- **Measurable**    요구사항 만족 검증을 위한 방법, 수용 기준
- **Attainable**     기술적 타당성
- **Realizable**     자원, 인원, 기술에 대한 현실성
- **Traceable**     Conception-Specification-Design-Implementation-Test

## 4. 품질 요구 명세

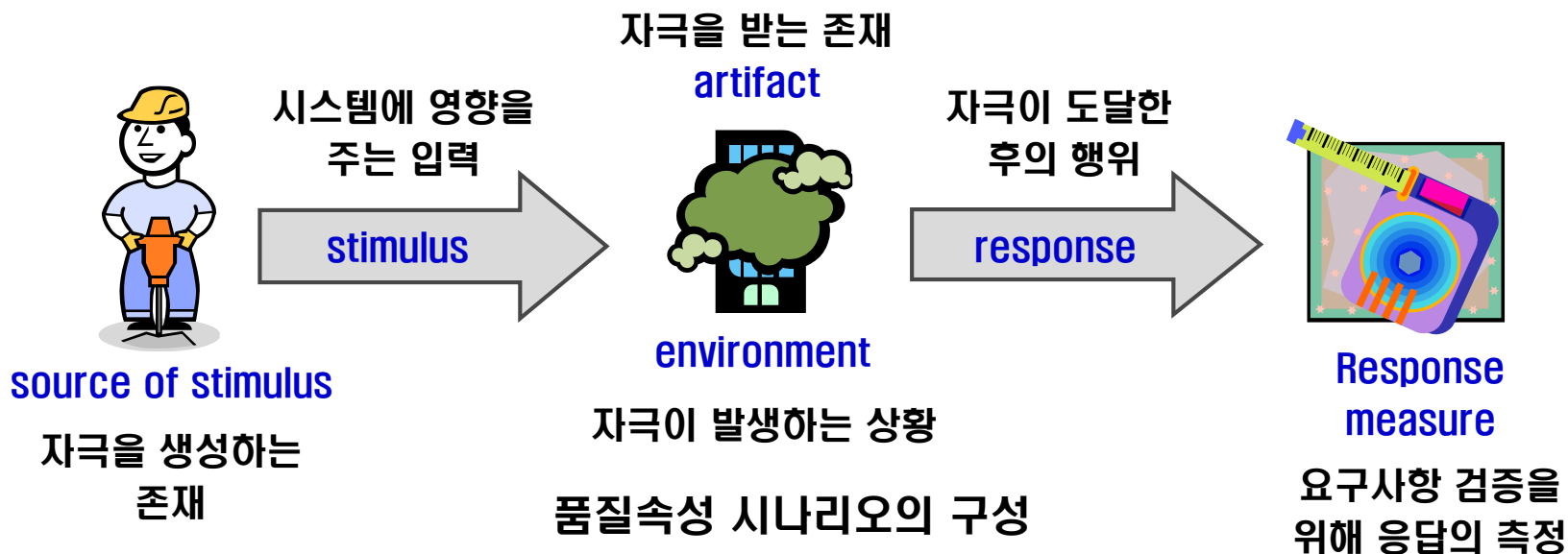
### Quality Attribute Scenario

#### 품질 속성 시나리오

- 품질 속성 기반 요구사항 명세 : 고객의 Needs로부터 품질속성을 쉽고 정확하게 추출
- 시스템마다 처리할 품질속성이 달라질 수 있다.

#### 품질속성 시나리오의 작성

- 시스템이 달성할 품질 속성의 목표 설정
- 시스템이 달성할 품질 속성을 식별



## 4. 품질 요구 명세

### Quality Attribute Scenario Template

<b>Requirement-ID</b>	QA_###
<b>Category</b>	관련된 Quality Attribute가 무엇인지 기술함 ( 예: Performance, Reliability, Security 등)
<b>Source</b>	Stimulus를 발생시키는 주체가 무엇인지 기술함
<b>Stimulus</b>	시스템에 입력되는 내외부 자극이 무엇인지 기술함
<b>Artifacts</b>	Stimulus의 영향을 받는 시스템의 내부 모듈, 컴포넌트, 혹은 시스템 전체
<b>Environment</b>	해당 Stimulus 발생시 시스템의 환경(운영모드일 수도 있고, 그 외 다양한 상황 가능)
<b>Response</b>	기술된 Environment에서 Artifact이 Stimulus를 받아들인 후 취하는 Action 이 무엇인지 설명함
<b>Response Measure</b>	위의 Response의 정도를 측정하는 단위가 무엇인지 기술함 (예: 초당 데이터 처리량, 반응시간, 시간일 경우 hour, minute, second 여부 등)
<b>Priority</b>	Quality Attribute Tree 상에서의 우선순위
<b>Description</b>	위에 기술된 Source부터 Response Measure까지의 내용을 하나의 문장으로 요약해서 기술함

## 4. 품질 요구 명세- 가용성 QAS 템플릿

### 가용성(Avalability) - 시스템 실패 및 결과와 관련

- 시스템이 명시한 서비스를 더 이상 제공하지 못할 때(실패 시) 발생
- 실패 보수 시간과 시스템이 필요할 때 동작하는 비율에 관심

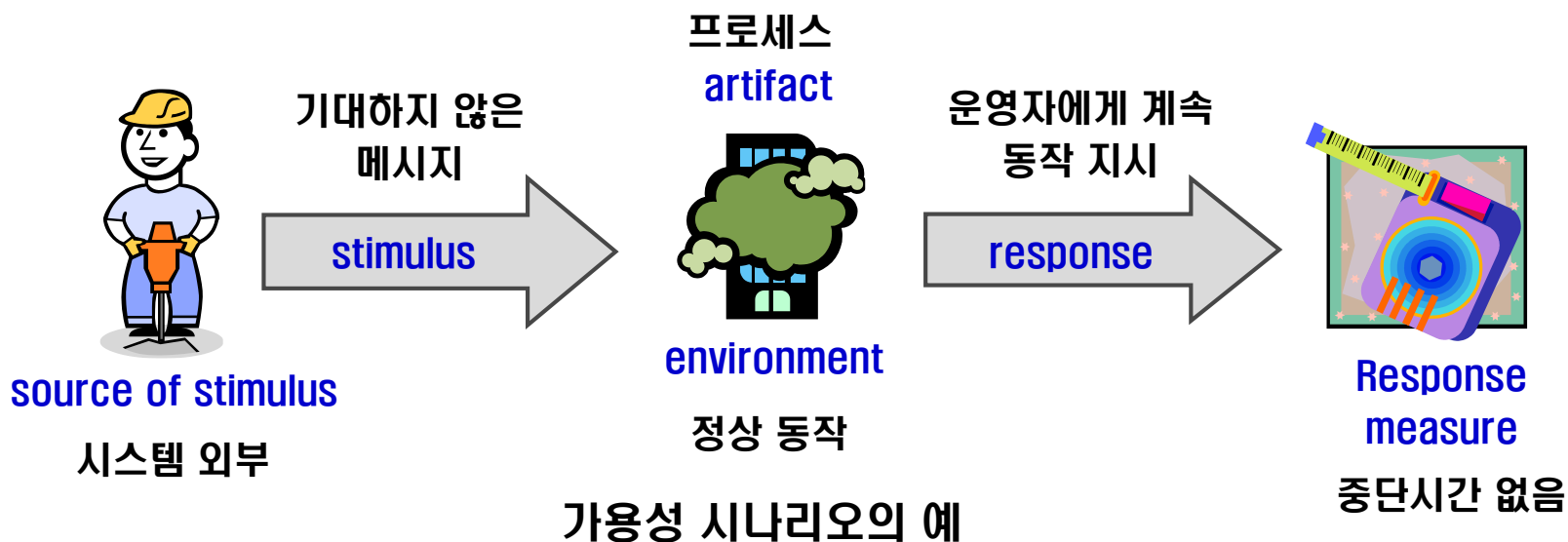
시나리오 항목	입력 가능 값
Source	•결함/실패의 원인 [시스템 내부/시스템 외부]
Stimulus	•발생 결함 [누락/정지/타이밍/응답]
Artifacts	•가용성을 위해 요구되는 자원 [프로세서/통신채널/자료저장소/프로세스]
Environment	•결함 또는 실패 발생의 시스템 상태 [정상 동작/저하 모드]
Response	<ul style="list-style-type: none"><li>• 실패를 기록한다</li><li>• 사용자 및 다른 시스템을 포함하는 적절한 당사자에게 알린다.</li><li>• 정의된 규칙에 따라 오류, 실패를 유발한 이벤트 소스를 비활성화시킨다</li><li>• 시스템 중요도에 따라 결정되는 지정된 간격에서 사용할 수 없게 한다</li><li>• 정상 모드 또는 성능 저하 모드에서 계속 작동한다</li></ul>
Response Measure	<ul style="list-style-type: none"><li>• 시스템이 사용 가능할 때까지의 시간 간격</li><li>• 가용성 시간</li><li>• 보수시간</li><li>•비정상모드에서 동작시간 간격</li></ul>

## 4. 품질 요구 명세

### 가용성 시나리오의 예

#### 가용성 시나리오 예)

“정상적인 동작상태에서 예상하지 못한 외부 메시지가 수신되면 프로세스는 메시지가 수신됐음을 운영자에게 알리고 진행 중이던 동작을 시스템 중단 없이 계속 수행한다.”





## 4. 품질 요구 명세 – 변경용이성 QAS 템플릿

### 변경용이성 – 변경 비용과 관련

- 무엇이 변경될 수 있는가? [대상체]
- 언제 변경이 일어나고 변경을 유발하는 것은 무엇인가? [환경]
- 변경사항 결정과 비용의 측정

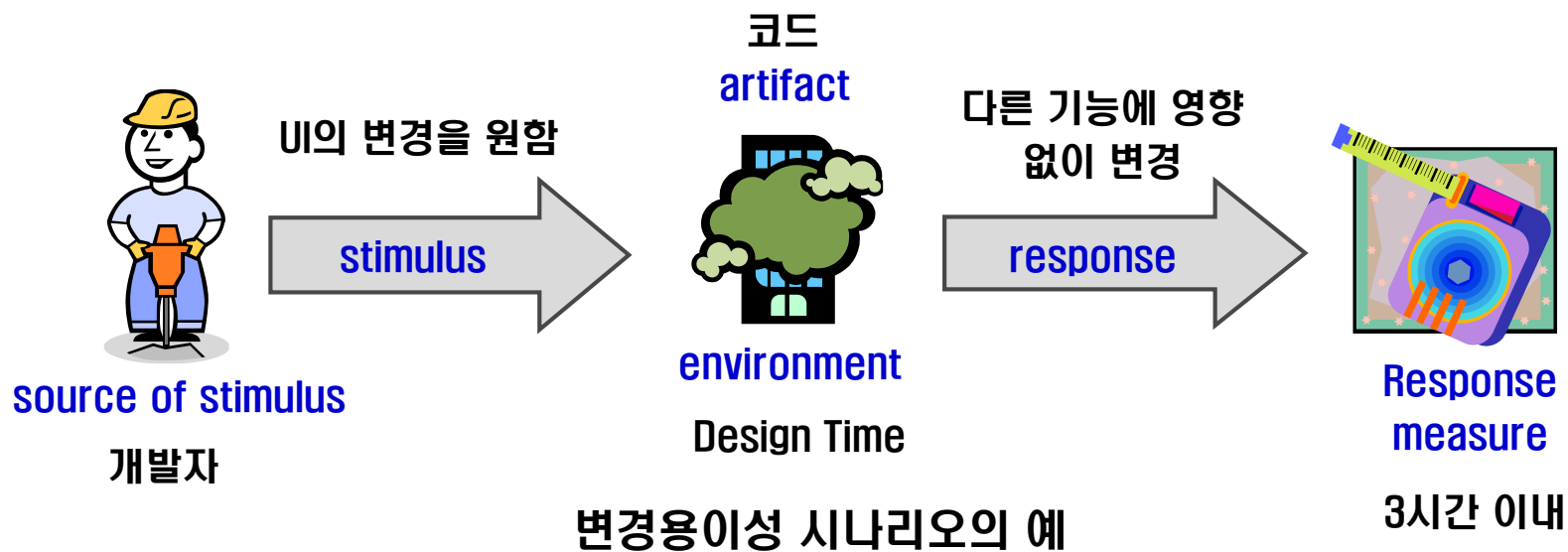
시나리오 항목	입력 가능 값
Source	• 변경 수행 인원 [최종사용자, 개발자, 시스템 관리자]
Stimulus	• 변경 내용 [기능 추가/수정/삭제, 품질속성 변경, 능력 변경]
Artifacts	• 변경될 대상 [시스템 기능성, UI, 플랫폼 환경, 다른 상호운용 시스템]
Environment	• 변경될 시점 [설계 단계, 컴파일 단계, 빌드 단계, 런타임 단계]
Response	<ul style="list-style-type: none"><li>• 아래의 내용에 해당하는 변경 수행 방법, 시험 방법, 배포 방법을 정하다</li><li>• 아키텍처 내의 변경위치를 지정함</li><li>• 다른 기능에 영향 없이 변경을 수행함</li><li>• 변경 결과를 테스트하고 결과를 배포함</li></ul>
Response Measure	• 응답 시간, 비용 [영향 받는 요소 수, 노력, 비용/다른 기능, 품질속성에 영향을 주는 범위]

## 4. 품질 요구 명세

### 변경용이성 시나리오의 예

#### 변경용이성 시나리오 예]

“개발자가 시스템의 사용자 인터페이스를 변경할 때 개발자는 설계 시점에서 코드를 시스템에 영향 없이 3시간 이내에 변경한다.”



## 4. 품질 요구 명세 - 성능 QAS 템플릿

### 성능 - 타이밍과 관련

- Event 발생부터 response 까지 소요 시간
- 성능 관련 요인 : Event 유발원, 이벤트 패턴과 응답 패턴
- 다중 사용자 : 도착 패턴의 다양화 모델링

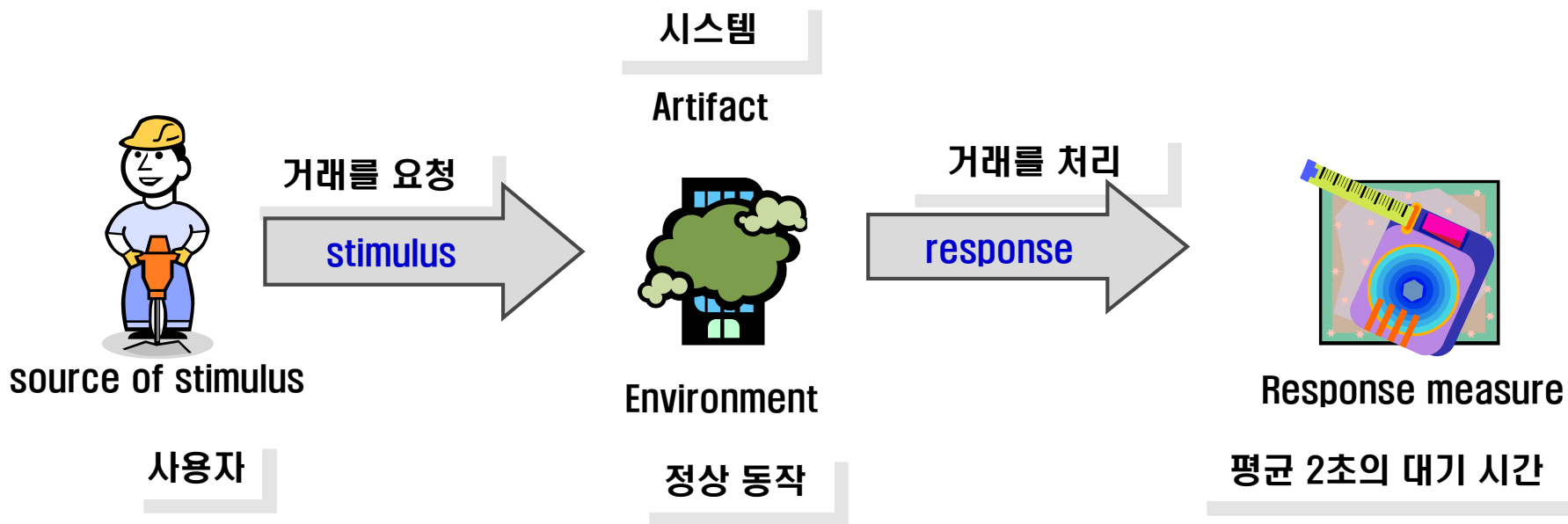
시나리오 항목	입력 가능 값
Source	<ul style="list-style-type: none"><li>• Event 발생 요인 (사용자 요청, 시스템 내부/외부)</li></ul>
Stimulus	<ul style="list-style-type: none"><li>• Event 도착 패턴 (Event의 주기적 도착, 산발적 도착, 확률적 도착)</li></ul>
Artifacts	<ul style="list-style-type: none"><li>• 시스템의 서비스</li></ul>
Environment	<ul style="list-style-type: none"><li>• 다양한 동작 모드 (정상 모드, 긴급 모드, 과부하 모드)</li></ul>
Response	<ul style="list-style-type: none"><li>• 도착한 이벤트의 처리하며 다음 내용을 결정한다</li><li>• 자극을 처리한다</li><li>• 서비스 수준을 변경한다</li></ul>
Response Measure	<ul style="list-style-type: none"><li>• 도착 이벤트의 처리시간/양 (대기시간, 처리마감시간, 지연시간, 분실률, 데이터 손실)</li></ul>

## 4. 품질 요구 명세

### 성능 시나리오의 예

#### 성능 시나리오 예)

“정상모드에서 사용자는 확률적으로 분당 1000개의 거래를 요청하며, 거래는 평균 2초의 대기 시간 내에 처리된다.”



## 4. 품질 요구 명세 – 보안성 QAS 템플릿

### 보안성 – 인가 받지 못한 사용의 거부와 관련

- 부인방지(non-repudiation)
- 기밀성(confidentiality)
- 무결성(integrity)
- 보증(assurance)
- 가용성(availability)
- 감사(auditing)

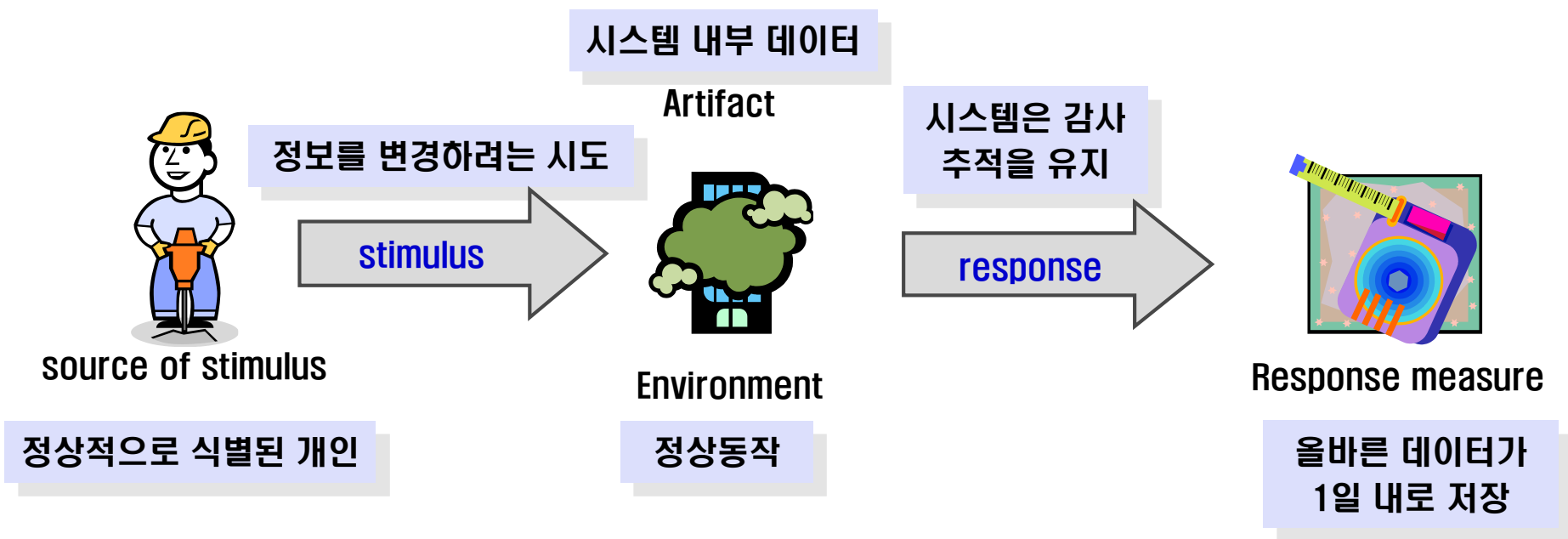
시나리오 항목	입력 가능 값
Source	• 공격 출처 (올바로 식별된/잘못 식별된/미확인 사람/다른 시스템 )
Stimulus	• 공격자체/보안침해시도 (데이터 접근/변경/삭제, 시스템 서비스 접근 시도, 시스템 가용성 절하)
Artifacts	• 공격 목표물 (시스템 서비스, 시스템 내부 데이터)
Environment	• Online/Offline (네트워크 연결/단절 상태, 방화벽/공개)
Response	<ul style="list-style-type: none"><li>• 사용자의 신원을 숨긴다</li><li>• 데이터/서비스에 대한 액세스를 (차단/ 허용) 한다.</li><li>• 데이터 /서비스에 액세스 할 수 있는 권한을 (부여/철회)한다</li><li>• 신원 확인을 통해 데이터/서비스에 대한 (접근/수정) 시도를 기록한다.</li><li>• 데이터를 읽을 수 없는 형식으로 저장한다.</li><li>• 서비스 접근을 감사(audit)하고 서비스 가용성을 제한합니다</li></ul>
Response Measure	• 공격을 막고 공격으로부터 복구에 드는 노력 (보안 측정에 필요한 시간/노력/자원, 공격 탐지 확률, 공격주체 식별 확률, 서비스 거부 공격 하에 사용 가능한 서비스 비율, 데이터/서비스의 복원, 손상된 데이터/서비스의 범위, 거부된 접근을 적법화하는 범위)

## 4. 품질 요구 명세

### 보안성 시나리오의 예

#### 보안성 시나리오 예]

“올바르게 식별된 개인 사용자가 시스템 사용자를 외부 지역에서 변경하려고 시도할 때 시스템은 감사 추적을 시작하고 하루 이내에 현재의 데이터를 복구한다.”



## 4. 품질 요구 명세 – 시험용이성 QAS 템플릿

### 시험용이성 – Fault 탐지가 용이한 시스템과 관련

- 확률로 정의 : 응답 측정치를 구한 후 다른 측정치를 사용
- 컴포넌트의 내부 상태와 입력 제어/출력 값 관찰이 가능하여야 함
- Test Harness 사용 : 코드, 설계 최종 시스템 항목

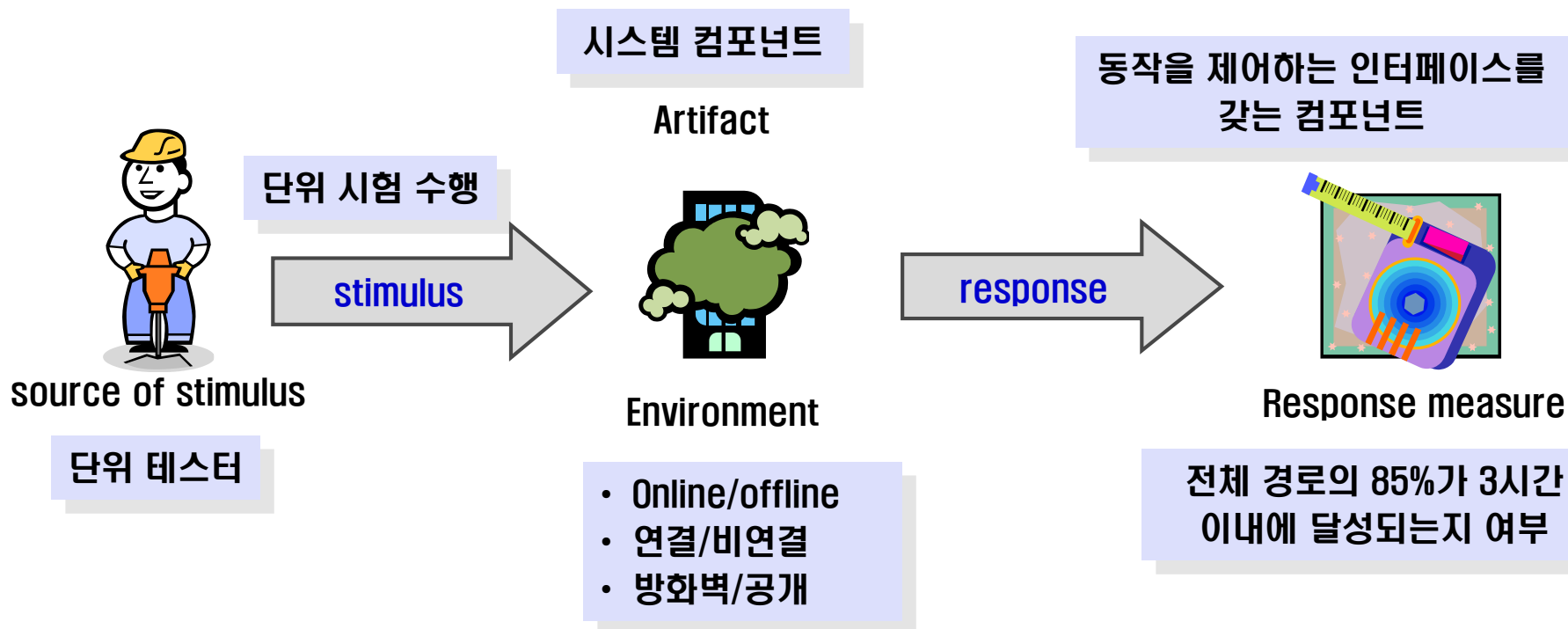
시나리오 항목	입력 가능 값
Source	• 단위 시험 담당자, 통합 시험 담당자, 시스템 시험 담당자, 고객 승인 담당자
Stimulus	• 개발 프로세스의 milestone (분석/아키텍처/설계/단위 클래스 코드 완성, 서브 시스템 통합 완료, 인도 시스템 완성)
Artifacts	• 시험 대상 (설계/소스코드 일부, 전체 시스템)
Environment	• 설계, 개발, 컴파일, 배치 시점
Response	• 응답 관찰 용이성/시험 제어용이성 (상태 값에 접근 가능하게, 연산된 값 제공, 시험 환경 준비)
Response Measure	• 실행 가능 구문 중 실제 실행 비율, 결함이 실패로 발생할 확률, 시험수행 시간, 시험 내 최장 의존 경로 길이, 시험 환경 준비 시간

## 4. 품질 요구 명세

### 시험용이성 시나리오의 예

#### 시험용이성 시나리오 예]

“단위 테스터는 개발이 완료된 시스템 컴포넌트에 대한 단위 시험을 수행한다. 이때의 컴포넌트는 자신의 행위를 제어하고 결과를 보여주기 위해 인터페이스를 제공한다. 그리고 전체 경로(path coverage)의 85%는 3시간 이내에 시험을 수행한다.”





## 4. 품질 요구 명세 – 사용용이성 QAS 템플릿

### 사용용이성 – 사용자의 작업 완료 노력과 시스템의 사용자 지원 종류와 관련

- 시스템 사양을 학습
- 시스템을 사용자 요구에 맞춤
- 시스템의 효율적 사용
- 신뢰와 만족감 높임, 에러의 영향을 최소화

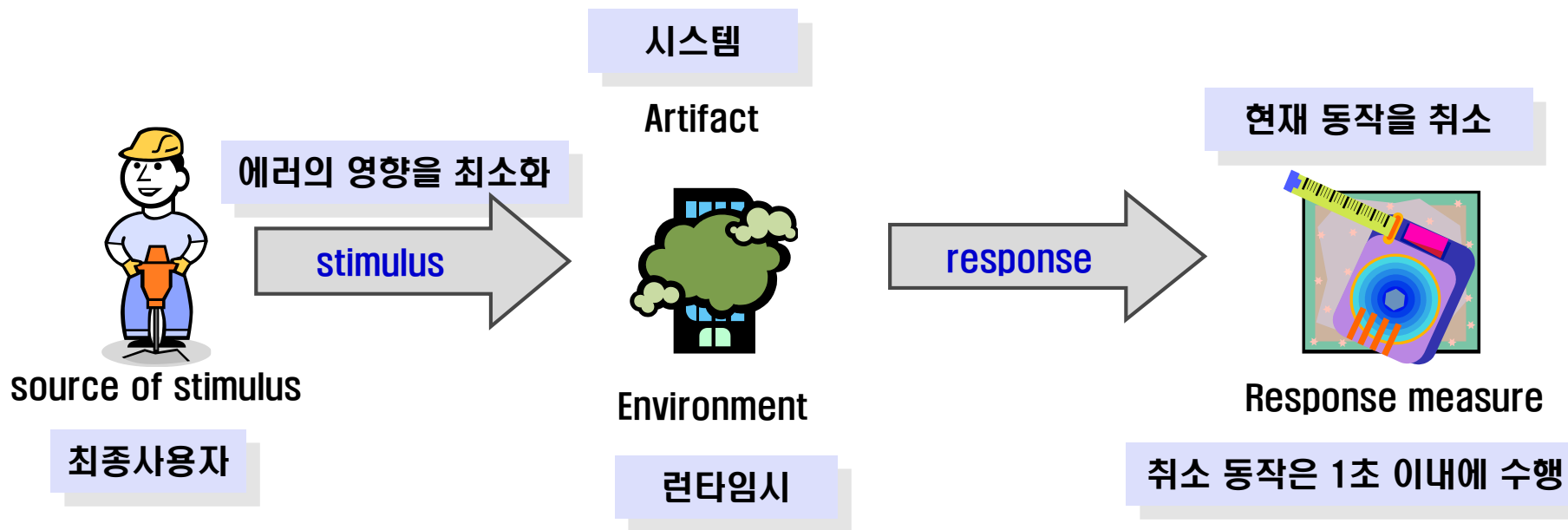
시나리오 항목	입력 가능 값
Source	<ul style="list-style-type: none"><li>• 최종 사용자</li></ul>
Stimulus	<ul style="list-style-type: none"><li>• 사용자가 시스템을 효율적 사용, 시스템 사용법 학습, 에러 영향 최소화, 시스템을 적합하게 변경, 사용에 편리함 요구</li></ul>
Artifacts	<ul style="list-style-type: none"><li>• 시스템</li></ul>
Environment	<ul style="list-style-type: none"><li>• 설계, 개발, 컴파일, 배치 시점</li></ul>
Response	<ul style="list-style-type: none"><li>• 시스템 기능 배우기 지원 상황에 민감한 도움말 시스템. 사용자에게 친숙한 인터페이스. 익숙하지 않은 환경에서의 사용; 이미 입력 된 데이터/명령의 재사용; 화면 내에서 효율적인 네비게이션 지원 등</li><li>• 오류의 영향 최소화 실행 취소, 취소, 시스템 오류 복구 관련 인터페이스를 사용할 수 있다.</li><li>• 효율적인 시스템 사용 지원 데이터 / 명령의 사용자 오류 인식 및 수정, 잊어 버린 암호 검색, 시스템 자원 확인</li><li>• 적응 시스템: 맞춤 설정; 국제화</li><li>• 편안한 느낌 : 디스플레이 시스템 상태, 사용자의 페이스로 작업</li></ul>
Response Measure	<ul style="list-style-type: none"><li>• 작업시간, 에러 수, 해결된 문제 수, 사용자 만족도, 성공 동작의 비율, 에러 발생 시 손실된 시간/데이터의 양</li></ul>

## 4. 품질 요구 명세

### 사용용이성 시나리오의 예

#### 사용용이성 시나리오 예)

“에러의 영향을 최소화하길 원하는 사용자가 런타임 시 시스템 동작을 취소하는 기능을 수행할 때 취소 동작은 1초 이내에 이루어져야 한다.”



## 5. 품질 시나리오와 아키텍처

### Quality Scenario and Architecture Evaluation

- ❑ Most importantly: gives information
  - Is the architecture suitable for the system for which it was devised?
  - Which of two competing architectures is most suitable for the system at hand?
- ❑ Architecture is suitable if
  - The system that results from it will meet its quality goals
  - The system can be built using the resources at hand (architecture is buildable)

## 5. 품질 시나리오와 아키텍처

### Quality Scenario and Architecture Evaluation

- ❑ Architecture suitable with respect to
  - A system is modifiable or not wrt a specific kind of change
  - A system is secure or not wrt a specific kind of threat
  - A system is reliable or not wrt a specific kind of fault occurrence
  - A system performs well or not wrt specific performance criteria
  - An architecture is buildable or not wrt specific time and budget constraints
- ❑ Questioning techniques
  - Rely on thought experiments to check architecture suitability
  - **Scenario-based style: ATAM**
  - Checklist-based style

## 5. 품질 시나리오와 아키텍처

### ATAM (Architecture Tradeoff Analysis Method)

- ❑ To assess the consequences of architectural decisions in light of quality attribute requirements
- ❑ Primarily a risk identification mechanism (p.60)
  - Alternatives that might create (future) problems in some quality attribute
- ❑ Discover sensitivity points
  - Alternatives for which a slight change makes a significant difference in a quality attribute
- ❑ Discover tradeoffs
  - Decisions affecting more than one quality attribute
- ❑ Side-effect of ATAM
  - Improve the architecture documentation
  - Foster stakeholder communication & consensus

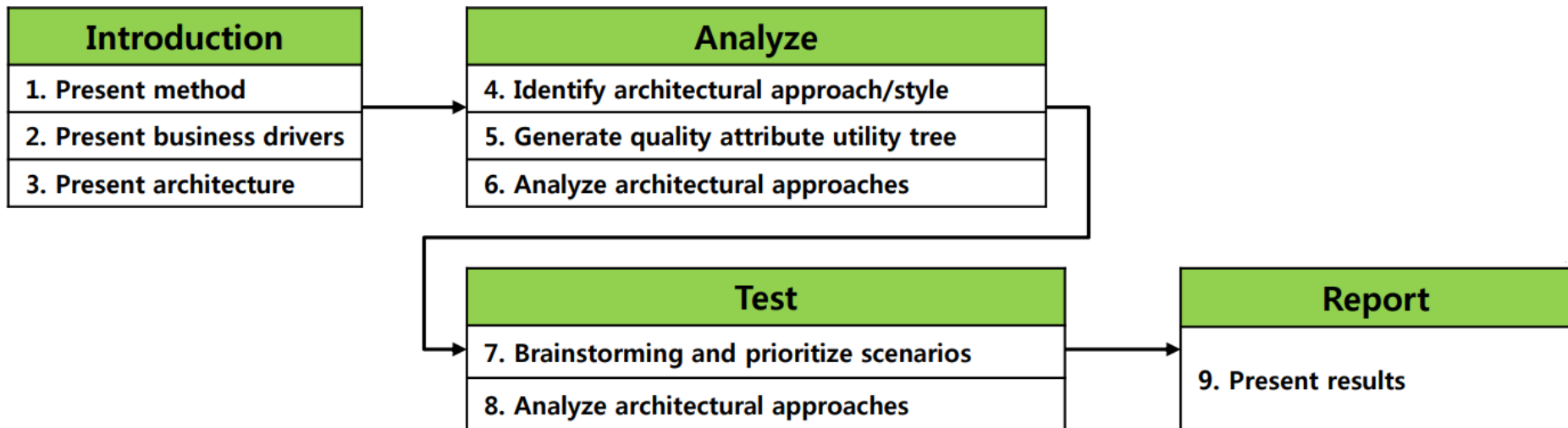
# 5. 품질 시나리오와 아키텍처

## ATAM 프로세스의 구성

### ❑ SEI Quality Attribute Workshop

- system-centric
- stakeholder focused
- scenario based

### ❑ ATAM Steps



# 5. 품질 시나리오와 아키텍처

## ATAM step - Present ATAM & Present Business Drivers

- ❑ Provides ATAM steps in brief
- ❑ ATAM customer representative describes the system's business drivers including:
  - Business context for the system
  - High-level functional requirements
  - High-level quality attribute requirements
  - Architectural drivers: quality attributes that shape the architecture
  - Critical requirements: quality attributes most central to the system's success

# 5. 품질 시나리오와 아키텍처

## ATAM step - Present & Identify Architecture

- ❑ **Architect presents an overview of the architecture** including (for example)
  - Technical constraints such as an OS, hardware, or middle-ware prescribed for use
  - Other systems with which the system must interact
  - Architectural approaches/styles used to address quality attribute requirements
- ❑ **Identify any predominant architectural approaches** (for example)
  - client-server
  - 3-tier
  - proxy
  - publish-subscribe
  - redundant hardware



# 5. 품질 시나리오와 아키텍처

## ATAM step - Generate Utility Tree

- ❑ Find the most important quality attribute and set priority by evaluation team and decision maker and share the priority of quality attribute
- ❑ Generate documents for quality attribute requirement and priority
- ❑ **Utility Tree**
  - Utility is the qualities are supported by system
  - Utility → Quality Attribute → Detailed Quality Attribute → Scenario
  - helpful to make decision to set a priority of quality attribute requirement
  - Assign a priority score to each scenario
  - Matrix for priority decision
- ❑ **Scenario**
  - Describe how to interact between system and stakeholder
  - Consist of Stimuli, Response, and Environment

# 5. 품질 시나리오와 아키텍처

## ATAM System Quality Attribute

❑ Performance

❑ Availability

❑ Usability

❑ Security

End User  
view

❑ Maintainability

❑ Portability

❑ Reusability

❑ Testability

Developer  
view

❑ Time To Market

❑ Cost and Benefits

❑ Projected life ime

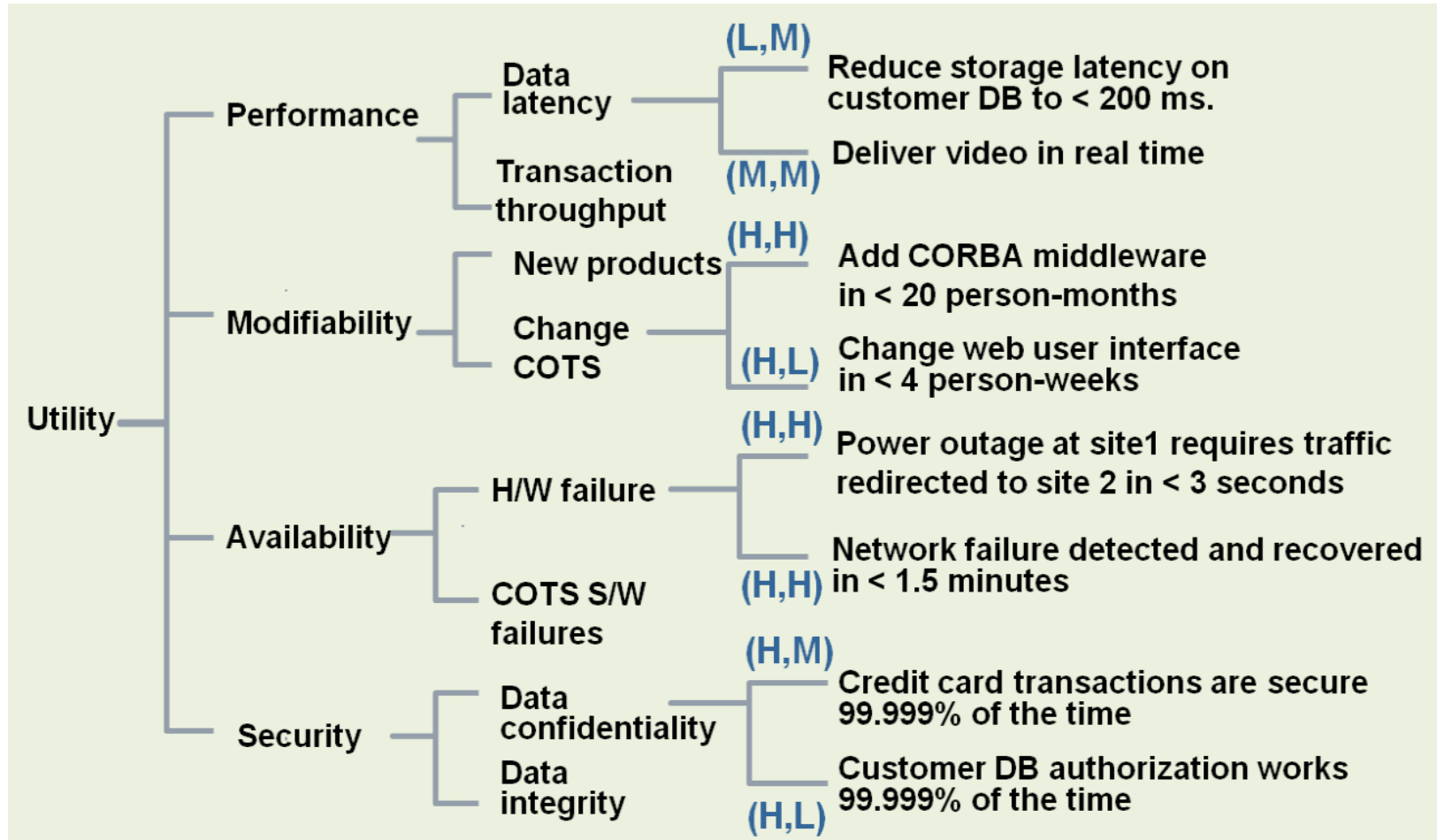
❑ Targeted Market

❑ Integration with  
Legacy System

Business  
Community  
view

# 5. 품질 시나리오와 아키텍처

## ATAM step - Generate Utility Tree



\*(Importance, Difficulty)

# 5. 품질 시나리오와 아키텍처

## ATAM step – Brainstorming & Analyze architectural approach

### ❑ Brainstorming and prioritize scenarios

- Stakeholders generate scenarios using a facilitated brainstorming process
- Examples are used to facilitate the step
- New scenarios are added to the leaves of the utility tree

### ❑ Analyze architectural approaches

- Identify the architectural approaches impacted by the scenarios generated in the previous step
- Continue identifying risks and non-risks
- Continue annotating architectural information

# 5. 품질 시나리오와 아키텍처

Analysis of Architectural Design				
Scenario #: Number	Scenario: Text of scenario from utility tree			
Attribute(s)	Quality attribute(s) with which this scenario is concerned			
Environment	Relevant assumptions about the environment in which the system resides, and the relevant conditions when the scenario is carried out			
Stimulus	A precise statement of the quality attribute stimulus (e.g. function invoked, failure, threat, modification ...) embodied by the scenario			
Response	A precise statement of the quality attribute response (e.g. response time, measure of difficulty of modification)			
Architectural Decisions	Sensitivity	Trade-off	Risk	Non-risk
Architectural design decisions relevant to this scenario that affect quality attribute	Sensitive point #	Trade-off point #	Risk #	Non-risk #
...	...	...	...	...
...	...	...	...	...
Reasoning	Qualitative and quantitative rationale for why the list of architectural decision contributes to meeting each quality requirement expressed by the scenario			
Architectural Diagram	Diagram or diagrams of architectural views annotated with architectural information to support the above reasoning, accompanied by explanatory text if desired			

- ❑ identifying sensitivity and tradeoff points
- ❑ backup database for reliability is a sensitivity point
- ❑ backup database has a tradeoff for reliability and performance
- ❑ backup database can be a risk depending on whether its performance cost is excessive in near future

# 5. 품질 시나리오

## Example of Analysis

- Scenario:  
Detect and recover from HW failure of main switch

Scenario #: A12		Scenario: Detect and recover from HW failure of main switch.			
Attribute(s)	Availability				
Environment	Normal operations				
Stimulus	One of the CPUs fails				
Response	0.999999 availability of switch				
Architectural decisions		Sensitivity	Tradeoff	Risk	Nonrisk
Backup CPU(s)		S2		R8	
No backup data channel		S3	T3	R9	
Watchdog		S4			N12
Heartbeat		S5			N13
Failover routing		S6			N14
Reasoning	<p>Ensures no common mode failure by using different hardware and operating system (see Risk 8)</p> <p>Worst-case rollover is accomplished in 4 seconds as computing state takes that long at worst</p> <p>Guaranteed to detect failure within 2 seconds based on rates of heartbeat and watchdog</p> <p>Watchdog is simple and has proved reliable</p> <p>Availability requirement might be at risk due to lack of backup data channel ... (see Risk 9)</p>				
Architecture diagram	<pre>graph LR; In(( )) --&gt; P[Primar CPU OS1]; In --&gt; B[Backup CPU with Watchdog OS2]; P --&gt; S[Switch CPU OS1]; B -- heartbeat 1 sec. --&gt; P; B --&gt; S; S --&gt; Out(( ))</pre>				

# 5. 품질 시나리오와 아키텍처

## ATAM – Step : Brainstorm & Prioritize Scenarios

- ❑ Stakeholders generate scenarios using a facilitated brainstorming process
  - Scenarios at the leaves of the utility tree serve as examples to facilitate the step
  - The new scenarios are added to the utility tree
- ❑ **ATAM Results include:**
  - Architectural approaches
  - Utility tree
  - Scenarios
  - Risks and non-risks
  - Sensitivity points and tradeoffs

# 예제 – Quality Attribute Scenario

## Virtual Shared Whiteboard Systems (VSWS)

### 1. Problem statement

**You are an architect for a company, Mobilosity.com, which produces products that support distributed meetings. Your job is to develop an architecture for a product that will allow a group of people to cooperate remotely using a virtual shared whiteboard. Each participant in the group should be able to draw on this shared whiteboard and see the results on whichever device they happen to be using. The initial offering should support various handheld devices, workstations, and eventually cell phones. Users should be able to join or leave a meeting at any time. The system should be able to store the results of a meeting, so that users can stop a meeting and then later start it up again.**



# **예제 – Quality Attribute Scenario**

## **Virtual Shared Whiteboard Systems (VSWS)**

### **2. Business Goal**

**To save cost and effort through time-free, device-free, place-free meetings.**

**To provide an easy way to initiate and conduct collaborative meeting.**

### **3 Constraints**

#### **– Business Constraints**

**The benefit of the system should exceed the cost**

**The architecture needs to be better than the other virtual meeting systems**

**Hand held devices should be easily connected to the system**

#### **– Technical Constraints**

**A network system must be available everywhere**

**The bandwidth of the system can accommodate multi-media data**

# 예제 – Quality Attribute Scenario

## Virtual Shared Whiteboard Systems (VSWS)

### ❑ Functional Requirements (sample)

- R1. VSWS shall provide multiple documents each of which could have multiple pages.
- R2. VSWS shall save and load documents in xml format.
- R3. VSWS shall export to jpeg.
- R4. VSWS shall change pen color and width.
- R5. VSWS shall navigate pages.
- R6. VSWS shall support basic command gesture (check for selection and scribble for deletion).
- R7. VSWS shall move and resize sketched figures.
- R8. VSWS shall provide freeform sketching.

# 예제 – Quality Attribute Scenario

## Virtual Shared Whiteboard Systems (VSWS)

### ❑ Architectural Requirement (sample)

- A1. VSWS shall accommodate different platforms. The architecture should provide a mechanism for applications to utilize all services. Any application can use any service.
- A2. Change in access devices should not modify the other devices' interface
- A3. To access VSWS is permitted only for authorized users.
- A4. VSWS shall provide for multiple users to access to the specific diagram.
- A5. VSWS shall provide a conference in real time.

# 예제 – Quality Attribute Scenario

## VSWS Utility Tree

Attribute	Concern	Scenario(s)	Related	Ranking	
			Req.	(I,D)*	Pri'ty
Interoperability	Infrastructure for Data exchanges	VSWS transforms the sketched graphical data into the XML document and exchanges it across the various platforms and devices.	A1	(H, M)	2
	Universal Use	Even if a user tries to access the white board by using their various handheld devices, VSWS provides same services by cellular phone adapter without any notification.	A1	(H, H)	1
Modifiability	Impact on other Components	Change of accessing devices should not modify the other devices' interface.	A2	(M, H)	3
Security	Auditing and logging	When a user tries to connect white board, VSWS always confirms whether he/she is approved user or not and maintains user connection information.	A3	(M, L)	6
Performance	Latency	Input data from one user can be transmitted to the other users within 5 seconds.	A5	(M, M)	5
	Capacity	Maximum number of concurrent users is 20 under guaranteed latency time.	A4	(M, H)	4

\*(Importance, Difficulty)

# 예제 – Quality Attribute Scenario

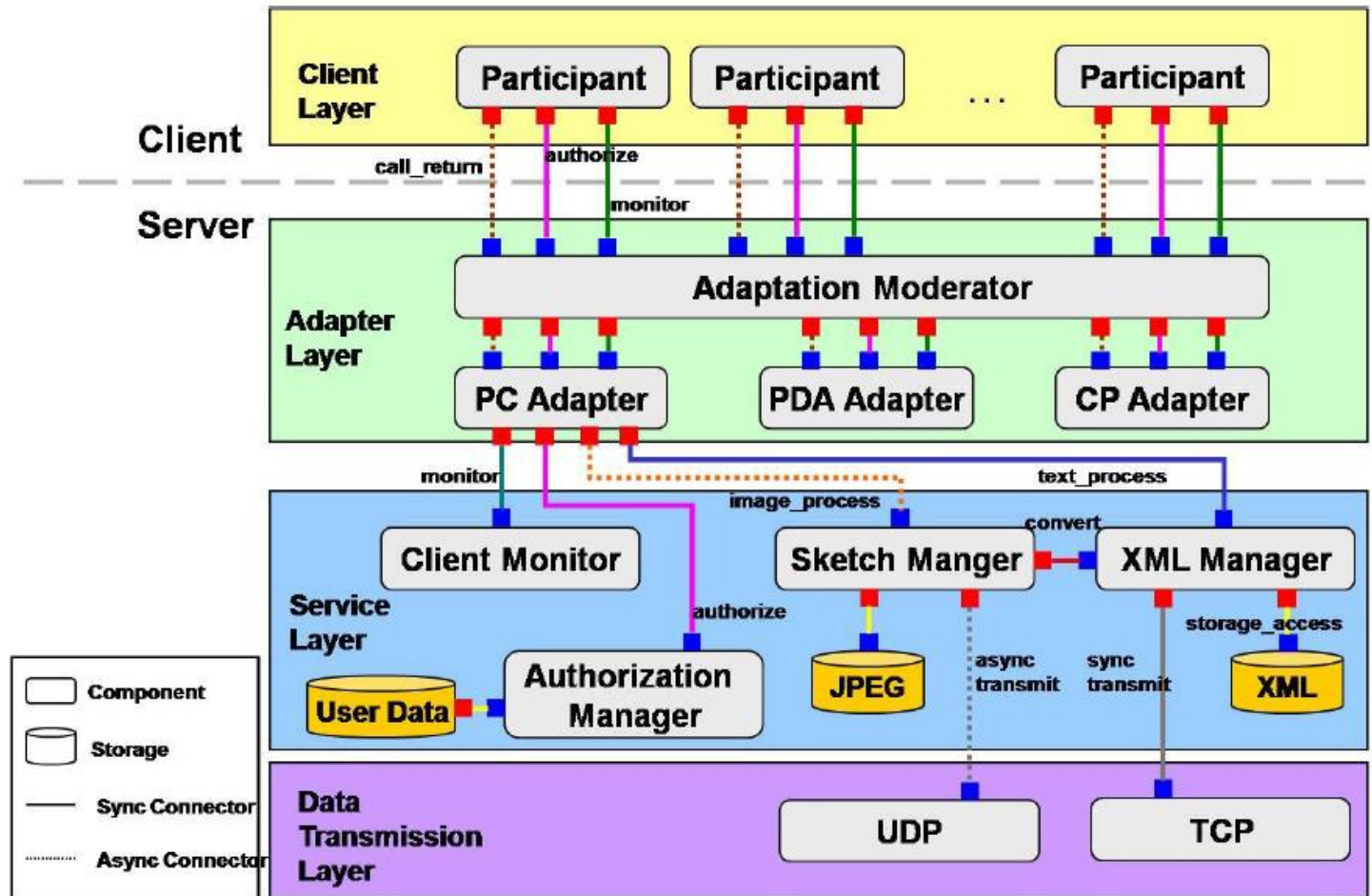
## QAS – Interoperability의 예

<b>Scenario</b>	1
<b>Name</b>	Transforming and exchanging sketched graphical data
<b>Attribute</b>	Interoperability – Infrastructure for data exchanges
<b>Environment</b>	Normal operations
<b>Stimulus</b>	Sketched graphical data
<b>Response</b>	Transform data into XML document Exchange it across the various platform and devices
<b>Response Measure</b>	Rate of successful format transformation, degree of user satisfactions, total number of format exchange errors
<b>Description</b> VSWS transforms the sketched graphical data into the XML document and exchanges it across the various platforms and devices	

<b>Scenario</b>	2
<b>Name</b>	Provide same service regardless of various handheld devices
<b>Attribute</b>	Interoperability – Universal use
<b>Environment</b>	Normal operations
<b>Stimulus</b>	Various user access
<b>Response</b>	Provide same services
<b>Response Measure</b>	Number of devices for interoperability, degree of user satisfactions, switching time intervals
<b>Description</b> Even if a user tries to access the whiteboard by using their various handheld devices, VSWS provides same services by cellular phone adapter without any notification..	

# 예제 – Quality Attribute Scenario

## VSWS Architecture Diagram



# 예제 – Quality Attribute Scenario

## VSWS Architecture - Adapter Layer : Indirect vs Direct

### ❑ Trade-offs : Interoperability(+) vs Performance(-)

- With current architecture, interoperability is increased because participant component does not need to know which specific device adapter is used
- Indirect communication through the adaptation moderator decreases performance of transaction

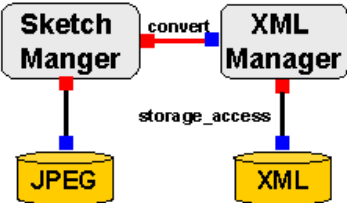

	Current : Indirect	Alternative1:Direct
C&C view		
+	Interoperability(S2)	Performance(S5)
-	Performance(S5)	Interoperability(S2)
Conclusion	Priority(Interoperability) > Priority(Performance)	

# 예제 – Quality Attribute Scenario

## VSWS Architecture - Data type : Hybrid data vs Mono Data

### ❑ Trade-offs : Interoperability(+) vs Performance(-)

- Only using image data has no problem in the PC or Laptop. But, for Cellular Phone, there is a limit of screen size and this limitation can be an obstacle in drawing some images.
- The XML format is recommended for interoperability across the divers devices. In the other hand, using the XML data causes the reduction of performance.

	Current : Hybrid Data	Alternative2 : Mono Data
C&C view		
+	Interoperability(S1)	Performance(S6)
-	Performance(S6)	Interoperability(S1)
Conclusion	Priority(Interoperability) > Priority(Performance)	



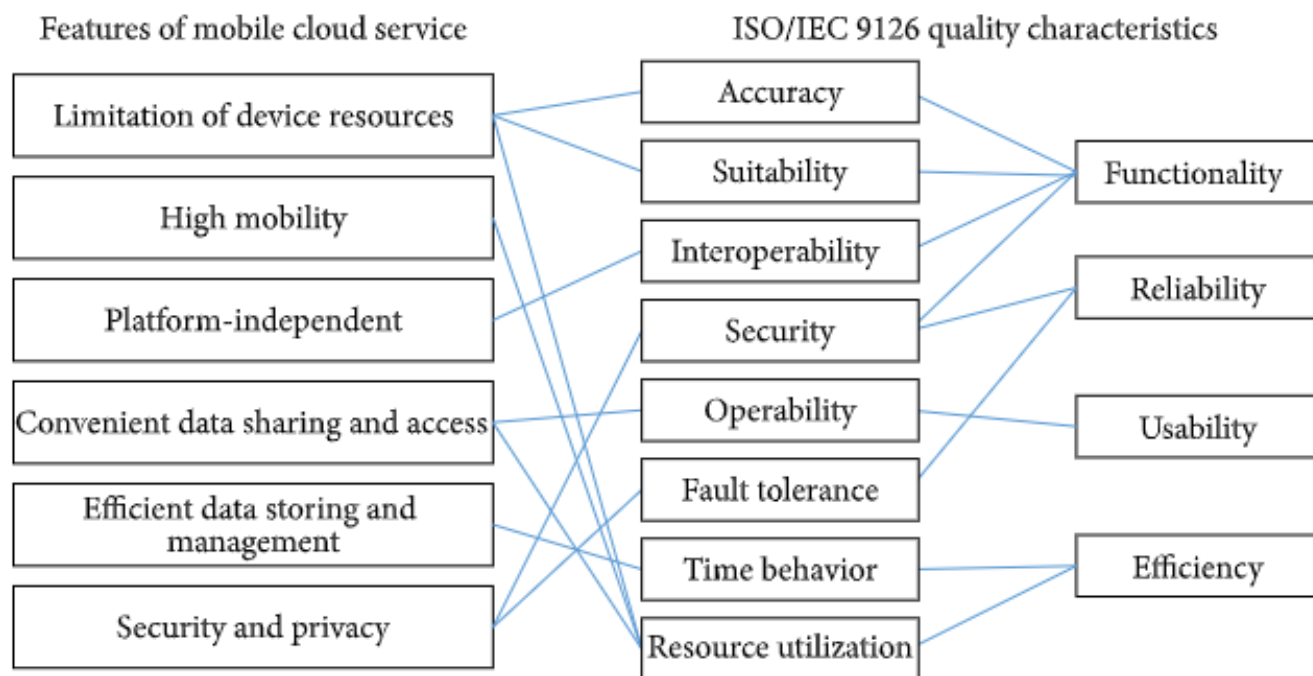
# Workshop #5 – 품질속성 정의

## step 1. 개인과제– 제품의 품질속성 선정

- QAS를 정의하기 앞서 각 팀별 case study 주제에 대해:

- 1) 중요하다고 생각되는 service feature 식별
- 2) IEC 9126 quality characteristics로 매핑

- 예: 모바일 클라우드 서비스의 경우



# Workshop #5 – 품질 속성 정의

## step 2. 개인과제 – Quality Attribute Scenario 작성

- ❑ 선정된 품질 속성 중 1가지를 선택하여 Quality Attribute Scenario를 작성합니다
- ❑ 예: 모바일 클라우드 서비스의 경우 Reliability, Usability 등
- ❑ 양식의 description항목에 품질속성 선정이유를 간단히 표기합니다.

- 시간: 개인과제 (60분)
- 포맷: 강의자료 69쪽 참조
- 평가주안점: 응용시스템의 특성이 QAS에 충분히 반영되어야 함
- 이메일 제출방법: **성명(팀명)\_QAS.xlsx** 혹은 **docs 링크**  
**(제출처:moberly@naver.com)**