Software Architecture Theory

A05. *More on Quality Attributes Design Strategies*

2014

Sungwon Kang

Table of Contents

1. 필요성

2. 품질 속성 설계 전략

3. 품질속성 설계 전략의 정의 절차

4. 전략 적용 사례

(This presentation combines Principles 2 and 10.)

1. 필요성

- **패턴(pattern)** : 반복적으로 발생하는 문제에 대한 미리 만들어놓은 솔루션으로 보통 여러 개의 힘의 충돌을 해결

- 패턴이 문제를 해결해 주지 못하면 어떻게 할 것인가?
  ☛ 품질속성 설계전략을 이용할 수 있다.
  - 품질속성 설계전략은 광범위하게 다룰 수 있는 해결의 틀을 제시한다.

- **품질속성 설계전략:** 단일 품질속성응답을 제어하는데 영향력 있는 설계결정

2. 품질 속성 설계 전략

- **What if you need more guidance than the six parts?**

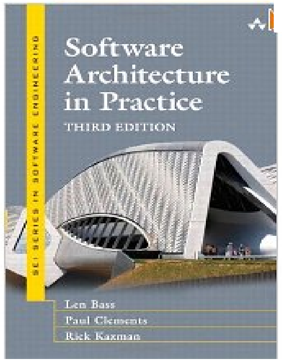
  ☛ *General scenarios* are those scenarios that are system independent.

  - represent quality attribute characterizations

- General scenarios can be used to create concrete scenarios which are specific to a particular system.

- General six-part scenarios exist for:

  (1) performance (성능)
  (2) modifiability (수정용이성)
  (3) availability (가용성)
  (4) usability (사용용이성)
  (5) interoperability (상호운영성)
  (6) testability (시험용이성)
  (7) security (보안성)


Software Architecture in Practice
THIRD EDITION
SEI SERIES IN SOFTWARE ENGINEERING
Len Bass
Paul Clements
Rick Kazman

[Bass 13]

(1) 성능 설계 전략(1/4)

- **Definition**: *Performance* is about timing: how long it takes the system to respond when an event occurs.
- **Areas of concern**:
  - **Sources of events** vary: interrupts, messages, requests from users, transactions, and so forth.
  - **Arrival rates and patterns** vary: sporadic, periodic, stochastic, or some combination.
  - **Response time**: elapsed time from event arrival to system reaction.

Based on [Kazman 04]

성능 설계 전략 (2/4)

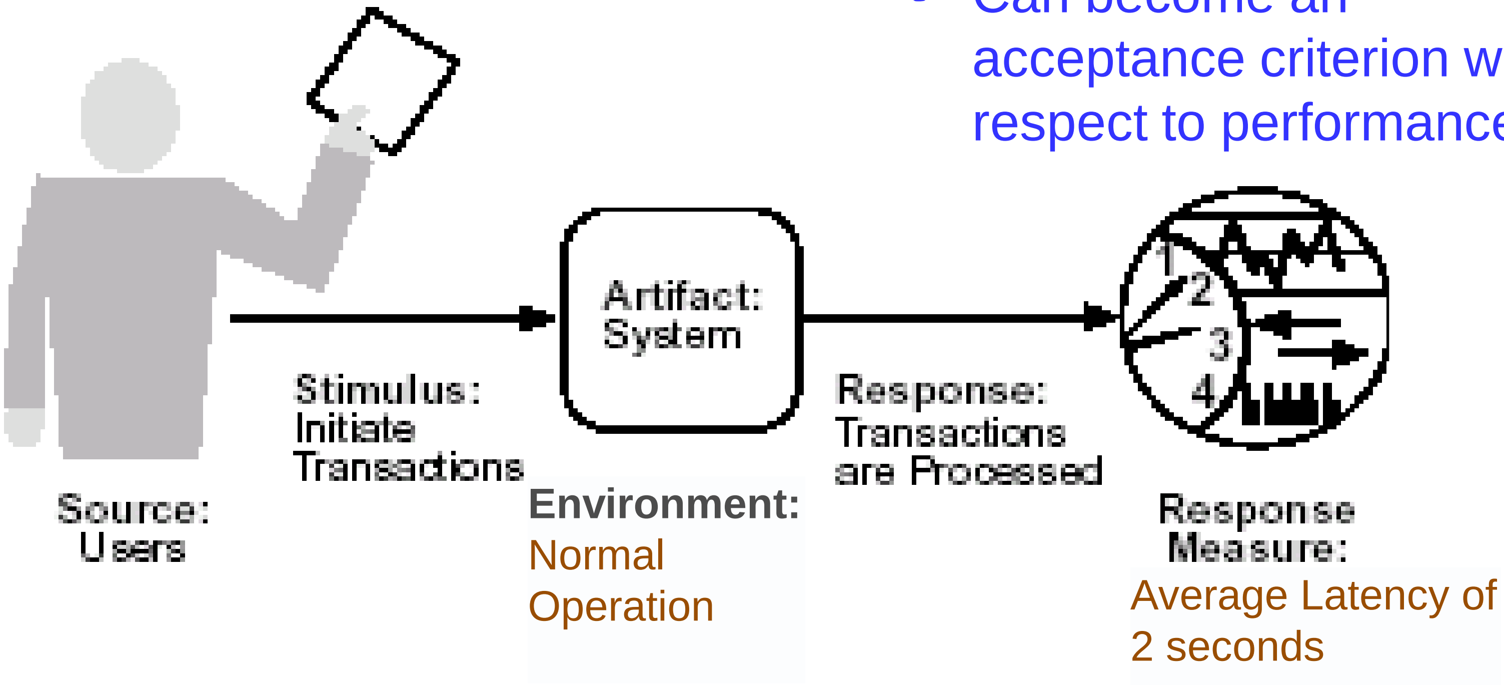
- **General Scenario Considerations**:

Source	Internal or external to the system
Stimulus	Arrival of a periodic, sporadic, or stochastic event
Artifact	**System or one or more components in the system**
Environment	Operational mode: normal, emergency, peak load, overload
Response	Processes events, change level of services
Response Measure	Latency, deadline, throughput, jitter, miss rate

성능 설계 전략 (3/4)

- 성능품질속성 시나리오 예 - *Users initiate 1000 transactions per minute stochastically under normal operations and these transactions are processed with an average latency of 2 seconds.*

☛ Can become an acceptance criterion with respect to performance



Environment: Normal Operation

Average Latency of 2 seconds

성능 설계 전략 (4/4)

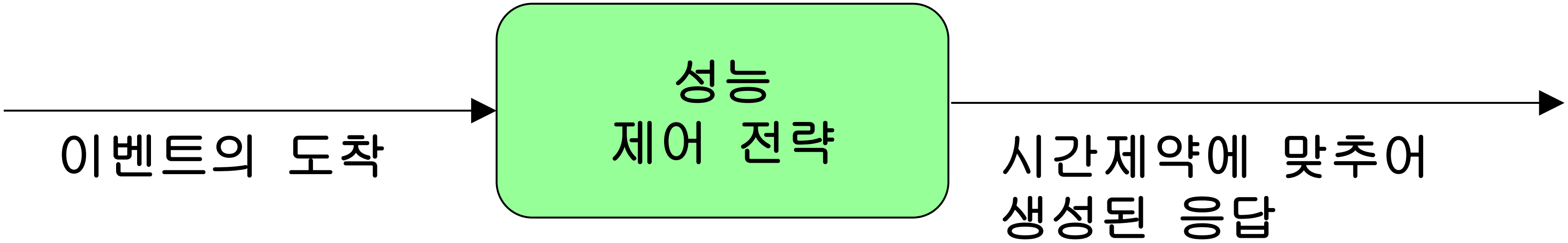


그림 8-3.성능 전략의 역할 (출처:[Bass 13]p.141)

자원 수용의 제어	자원의 관리
샘플링 비도 조절	가용 자원의 증대
이벤트 응답 속도 관리	병행성(concurrency) 도입
이벤트의 우선순위 부여	복수의 컴포넌트 복제본(copy) 유지
처리 오버헤드(overhead) 감소	복수의 데이터 복제본 유지
실행시간 상한선 설정	큐 크기의 상한선 설정
자원 효율성의 증대	자원의 스케쥴링

(2) 변경용이성 설계 전략 (1/4)

- **Definition**: *Modifiability* is about the cost of change and refers to the ease with which a software system can accommodate changes.

- **Areas of concern**:
  - Identify **what can change**
    - functions, platforms, hardware, operating systems, middleware, systems it must operate with, protocols, and so forth
    - quality attributes: performance, reliability, future modifiability, and so forth
  - **Whe**n will the change be made and **who** makes it?

Based on [Kazman 04]

변경용이성 설계 전략 (2/4)

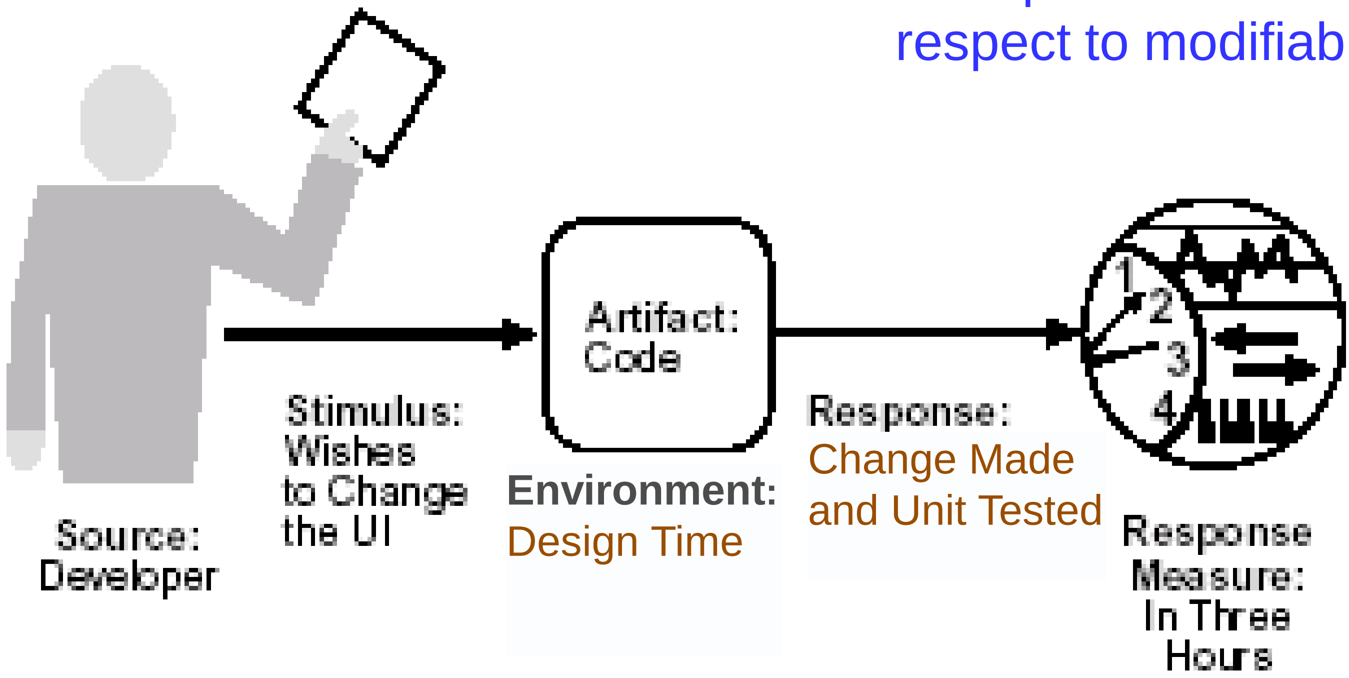
- **General Scenario Considerations**:

Source	End user, developer, system administrator
Stimulus	A directive to add/delete/modify functionality, or change a quality attribute, capacity, or technology
Artifacts	**Code, data, interfaces, components, resources, configuration, ...**
Environment	Run time, compile time, build time, initiation time, design time
Response	One or more of the following: - Make modification - Test modification - Deploy modification
Response Measure	Cost in terms of the following: - Number, size, complexity of affected artifacts - Effort - Calendar time - Money (direct outlay or opportunity cost) - Extent to which this modification affects other functions or quality attributes - New defects introduced

변경용이성 설계 전략 (3/4)

- 변경용이성 품질속성 시나리오 예 - *A developer wishes to change the UI code at design time. The modification is made with no side effects, in three hours*.

☛ Can become an acceptance criterion with respect to modifiability


Source: Developer
Stimulus: Wishes to Change the UI
Artifact: Code
Response:
Response Measure: In Three Hours
1
2
3
4

Environment: Design Time

Change Made and Unit Tested

변경용이성 설계 전략 (4/4)

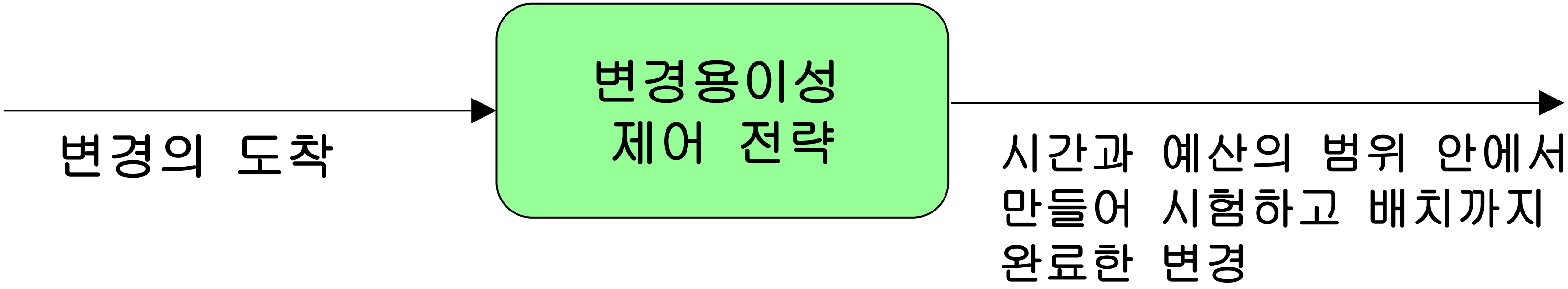


그림 7-2.변경용이성 전략의 역할 (출처:[Bass 13]p.121)

모듈크기 축소	응집도 증대	결합도 축소	바인딩의 지연
모듈의 분리	의미적 정합성 (coherence)의 증대	캡슐화 중개자(intermediary)의 사용 의존관계의 제한 리팩토링 공통서비스의 추상화	

(3) 가용성 설계전략 (1/4)

- **Definition**: *Availability* is concerned with system failure and its associated consequences. A system failure occurs when a system no longer delivers a service which is consistent with its specification.

- **Areas of concern**:
  - Preventing catastrophic system failure
  - Detecting system failure
  - Ability to recovering from system failure
  - Time to recover from system failure
  - Frequency of failure
  - Degraded modes of operation due to system failure

Based on [Kazman 04]

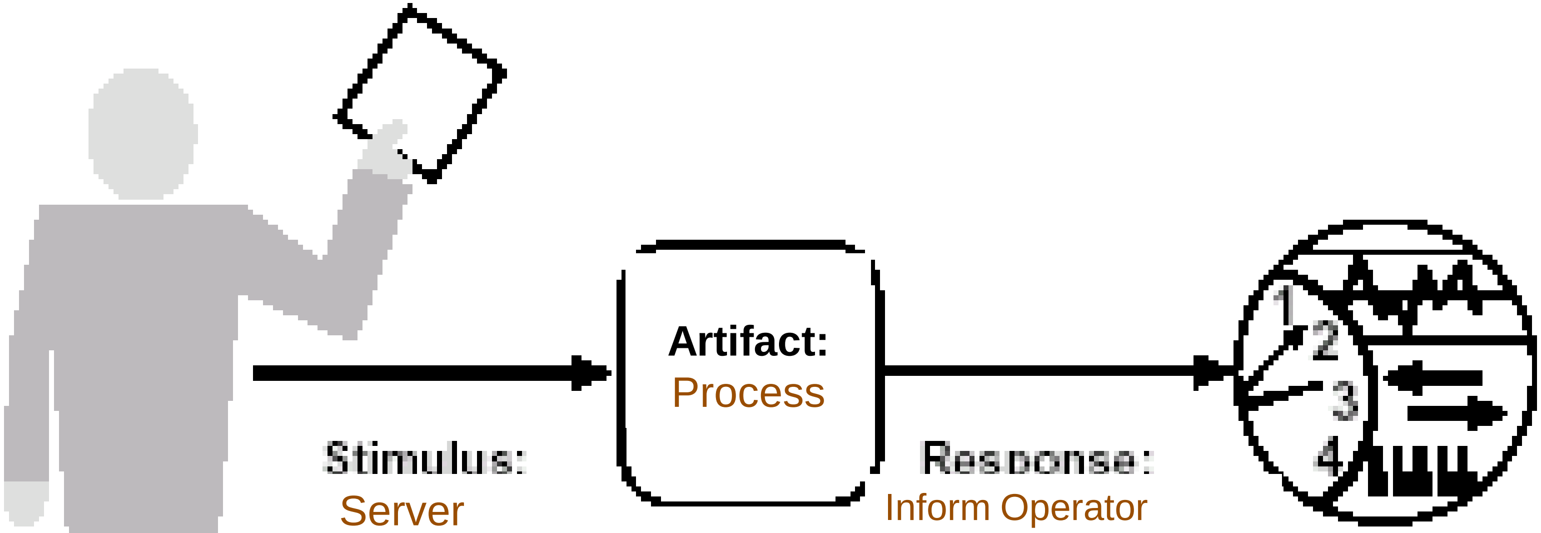
가용성 설계전략 (2/4)

- **General Scenario Considerations**:

Source	Internal/external: people, hardware, software, physical infrastructure, physical environment
Stimulus	Fault: omission, crash, incorrect timing, incorrect response
Artifacts	**Processors, communications channels, persistent storage, processes**
Environment	Normal operation, startup, shutdown, repair mode, degraded operation, overloaded operation
Response	Prevent the fault from becoming a failure Detect the fault: • Log the fault • Notify appropriate entities (people or systems) Recover from the fault: • Disable source of events causing the fault • Be temporarily unavailable while repair is being effected • Fix or mask the fault/failure or contain the damage it causes • Operate in a degraded mode while repair is being effected
Response Measure	Time or time interval when the system must be available Availability percentage(e.g., 99.999%) Time to detect the fault Time to repair the fault Time or time interval in which system can be in a degraded more Proportion(e.g., 99%) or rate(e.g., up to 100 per second) of a certain class of faults that the system prevents, or handles without failing

가용성 설계전략 (3/4)

- 가용성 품질속성 시나리오 예 - *An unanticipated external message is received by a process during normal operation. The process informs the operator of the receipt of the message and the system continues to operate with no down time.*



Source:
Heartbeat Monitor

Stimulus:
Server Unresponsive

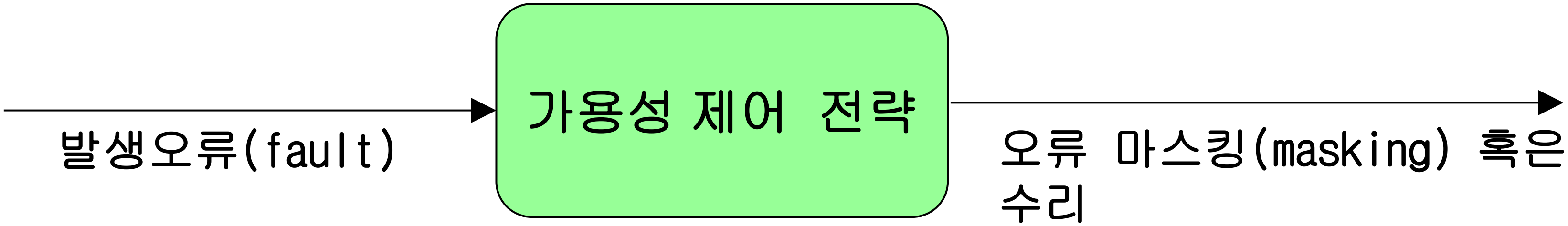
Artifact:
Process

Environment:
Normal Operation

Response:
Inform Operator
Continue to Operate

Response Measure:
No downtime

가용성 설계전략 (4/4)



발생오류(fault)

가용성 제어 전략

오류 마스킹(masking) 혹은 수리

그림 5-4. 가용성 전략의 목표 (출처:[Bass 13]p.177)

발생오류 탐지	오류로부터 복구		오류발생의 방지
	준비와 수리	재도입	
Ping/Echo 모니터 Heartbeat 타임스탬프 Sanity Checking 상태 모니터링 Voting Exception 탐지 Self-Test	Active Redundancy Passive Redundancy 여분 Exception 처리 Rollback SW 업그레이드 Retry 오류행위의 무시 Degradation 재구성	Shadow 상태 Resynch Escalating Restart Non-stop forwarding	서비스로부터 제거 트랜잭션 예측모델 Exception 방지 Competence Set의 증대

(4) 사용용이성 설계전략 (1/4)

- **Definition**: *Usability* is how easy it is for a user to accomplish a desired task and the kind of support the system provides for the user.

.

- **Areas of concern**:
  - Learning system features
  - Using a system efficiently
  - Minimizing the impact of errors
  - Adapting the system to user needs
  - Increasing confidence and satisfaction

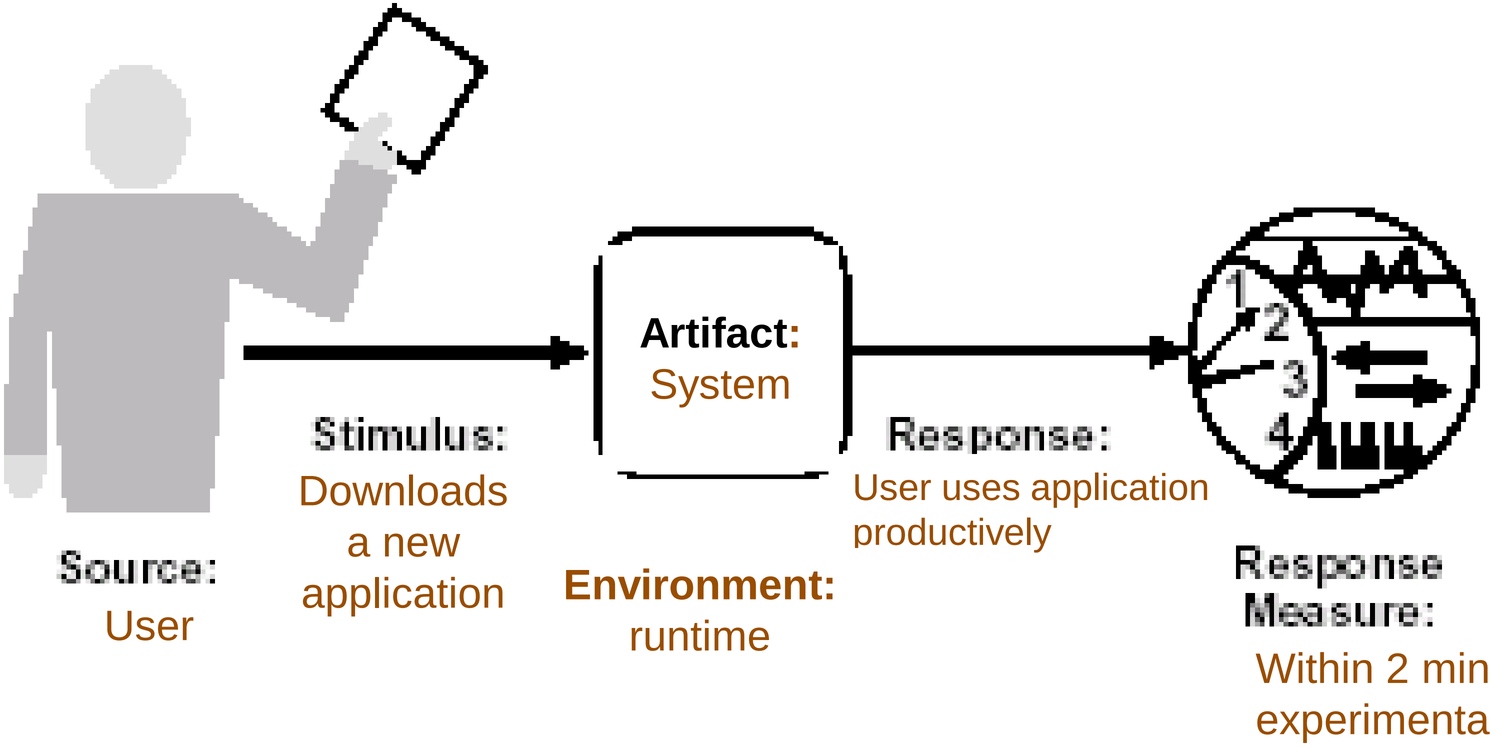
사용용이성 설계전략 (2/4)

- **General Scenario Considerations**:

Source	End user, possibly in a specialized role
Stimulus	End user tries to use a system efficiently, learn to use the system, minimize the impact of errors, adapt the system, or configure the system
Environment	Runtime or configuration time
Artifacts	**System or the specific portion of the system with which the user is interacting**
Response	The system should either provide the user with the features needed or anticipate the user's needs.
Response Measure	One or more of the following: task time, number of errors, number of tasks accomplished, user satisfaction, gain of user knowledge, ratio of successful operations to total operations, or amount of time or data lost when an error occurs

사용용이성 설계전략 (3/4)

- 사용용이성 품질속성 시나리오 예 - *A user, wanting to minimize the impact of an error, wishes to cancel a system operation at run time. The cancellation takes place in less than 1 second.*


Source:
User
Stimulus:
Downloads a new application
Artifact:
System
Environment:
runtime
Response:
User uses application productively
Response Measure:
Within 2 minutes of experimentation
1
2
3
4

사용용이성 설계 전략 (4/4)

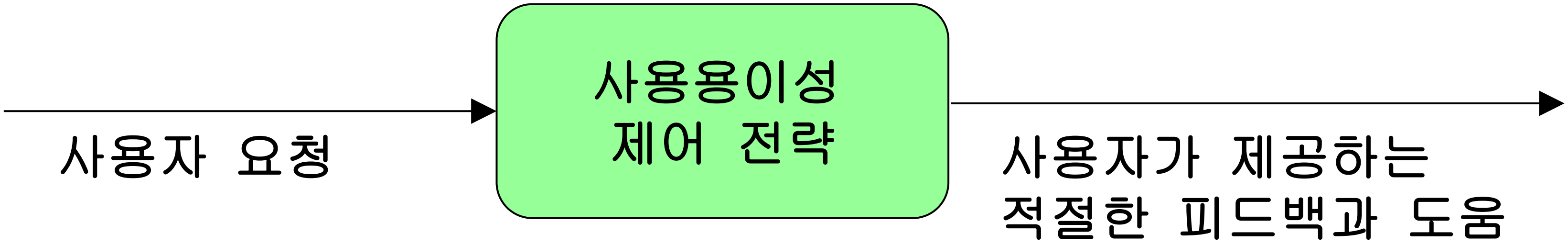


그림 11-2. 사용용이성 전략의 역할 (출처:[Bass 13]p.177)

사용자 동작 지원	시스템 동작 지원
취소	태스크 모델 유지
undo	사용자 모델 유지
멈춤/재개	시스템 모델 유지
반복적인 동작들을 묶는 동작제공	

(5) 상호운용성 설계전략(1/4)

- **Definition**: *Interoperability* is about the degree to which two or more systems can usefully exchange meaningful information via interfaces in a particular context. (<– Too narrow: ksw)

- **Areas of concern**:

  1) **Discovery**: The consumer of a service must discover (possibly at runtime, possibly prior to runtime) the location, identity, and the interface of the service.

  2) **Handling of the response**. There are three possibilities:

     - The service reports back to the requester with the response.
     - The service sends its response on to another system.
     - The service broadcasts its response to any interested parties.

Based on [Bass 13]

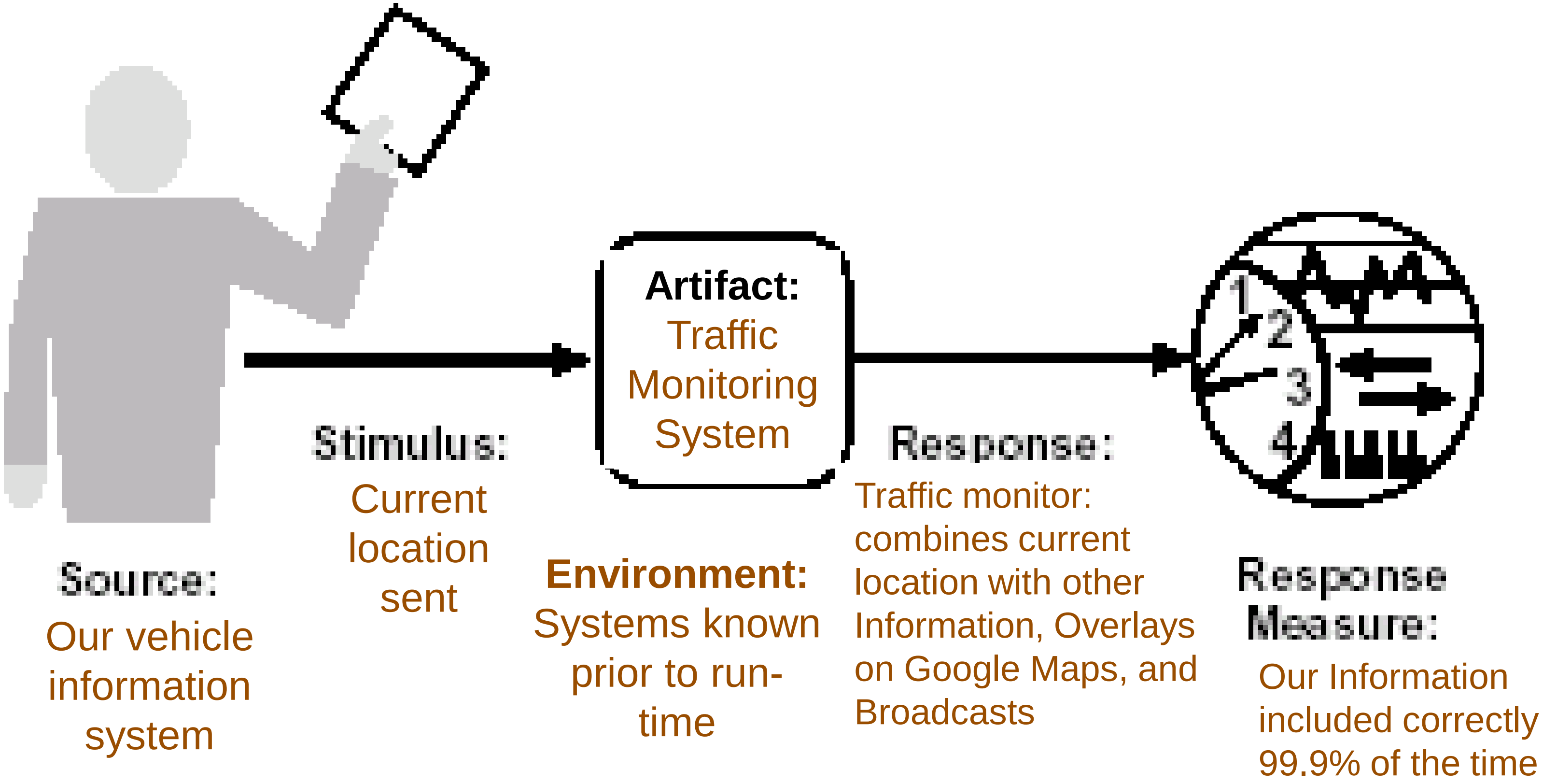
상호운용성 설계전략 (2/4)

- **General Scenario Considerations**:

Source	A system initiates a request to interoperate with another system
Stimulus	A request to exchange information among system(s)
Environment	System(s) wishing to interoperate are discovered at runtime or known prior to runtime
Artifacts	**The System that wishes to interoperate.** (ksw: should be 'systems' because it is interoperability between two ore more.)
Response	One or more of the following: • The request is (appropriately rejected and appropriate entities (people or system) are notified. • The request is (appropriately) accepted and information is exchanged successfully. • The request is logged by on e or more of the involved systems.
Response Measure	One or more of the following: • Percentage of information exchanges correctly processed • Percentage of information exchanges correctly rejected

상호운용성 설계전략 (3/4)

- 상호운용성 품질속성 시나리오 예 - *Our vehicle information system sends our current location to the traffic monitoring system. The traffic monitoring system combines our location with other information, overlays this information on a Google Map, and broadcasts it. Our location information is correctly included with a probability of 99.9%.*



Source:
Our vehicle information system

Stimulus:
Current location sent

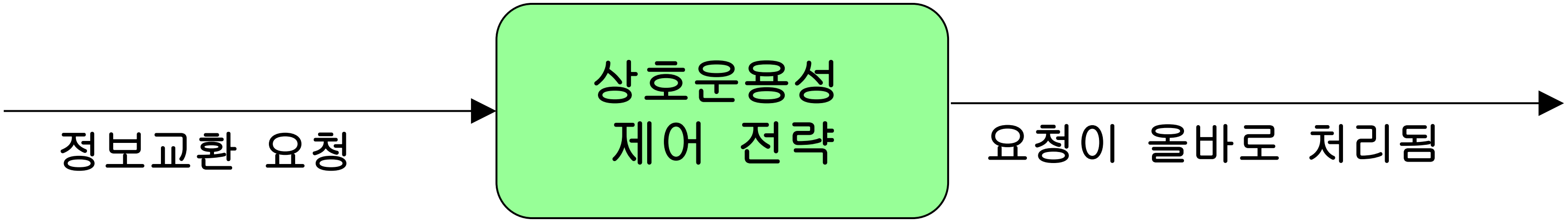
Artifact:
Traffic Monitoring System

Environment:
Systems known prior to run-time

Response:
Traffic monitor: combines current location with other Information, Overlays on Google Maps, and Broadcasts

Response Measure:
Our Information included correctly 99.9% of the time

상호운용성 설계전략 (4/4)


정보교환 요청

상호운용성 제어 전략

요청이 올바로 처리됨

그림 6-2. 상호운용성 전략의 목표 (출처:[Bass 13]p.110)

(서비스) 위치파악	인터페이스의 관리
서비스의 발견	오케스트레이트(Orchestration "scripts" interaction.) 인터페이스를 맞춤(tailor)

(6) 시험용이성 설계전략 (1/4)

- **Definition**: *Testability* is the ease with which the software can be made to demonstrate its faults through testing.

- **Areas of concern**:
  - On average 40% of the cost of development is taken up by testing.
  - For a system to be testable, it must be possible to control each component's internal state.

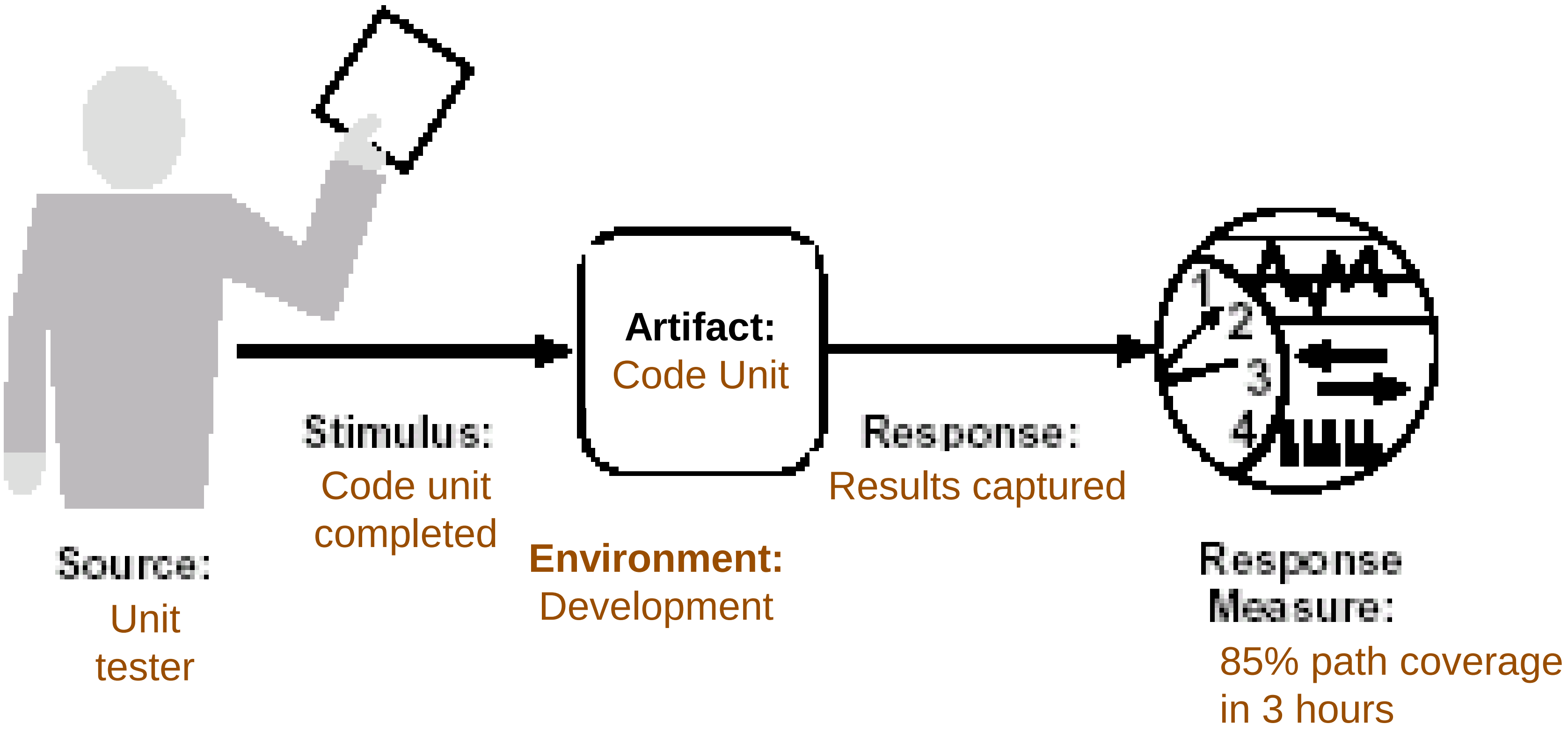
시험용이성 설계전략 (2/4)

- **General Scenario Considerations**:

Source	Unit testers, integration testers, system testers, acceptance testers, end users, either running tests manually or using automated testing tools
Stimulus	A set of tests is executed due to (1) the completion of a coding increment such as a class layer or service, (2) the completed integration of a subsystem, (3) the complete implementation of the whole system, or (4) the delivery of the system to the customer
Environment	Design time, development time, compile time, integration time, deployment time, run time
Artifacts	**The portion of the system being tested**
Response	One or more of the following: execute test suite and capture results, capture activity that resulted in the fault, control and monitor the sate of the system
Response Measure	One or more of the following: Effort to find a fault or class of faults, Effort to achieve a given percentage of state space coverage, probability of fault being revealed by the next test, time to perform tests, Effort to detect faults, length of longest dependency chain test, length of time to prepare test environment, reduction in risk exposure(size(loss)x prob(loss))

시험용이성 설계전략 (3/4)

- 시험용이성 품질속성 시나리오 예 - *A unit tester performs a unit test on a completed system component which provides an interface for controlling its behavior and observing its output. 85% path coverage is achieved within 3 hours.*



Source: Unit tester

Stimulus: Code unit completed

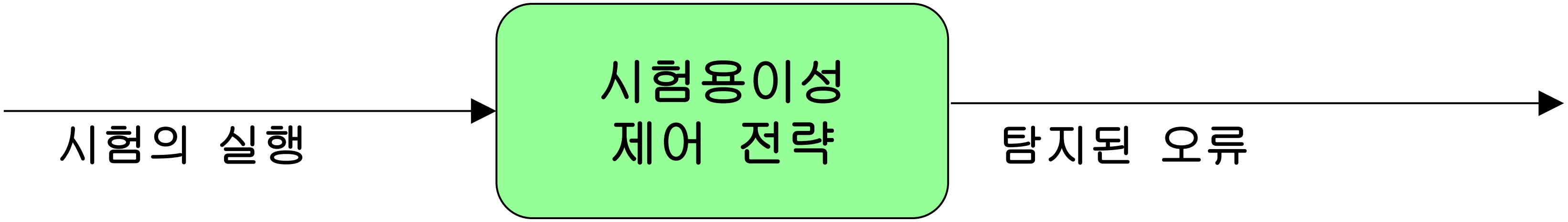
Artifact: Code Unit

Environment: Development

Response: Results captured

Response Measure: 85% path coverage in 3 hours

시험용이성 설계전략 (4/4)



시험의 실행

시험용이성 제어 전략

탐지된 오류

그림 10-3. 시험용이성 전략의 목표 (출처:[Bass 13]p.164)

시스템 상태를 제어관찰	복잡도 제한
특별한 인터페이스 기록/재생 상태 저장소를 국지화 또는 상태 기계를 사용 데이터 인터페이스를 추상화(국지화) 샌드박스(시스템 인스턴스를 고립화)	구조적 복잡도 제한 비결정성의 제한

(7) 보안성 설계전략 (1/5)

- **Definition**: *The measure of the system's ability to resist unauthorized attempts at usage (data or services) while providing access to legitimate users*
- **Areas of concern**:
  - Non-repudiation(부인방지): transactions cannot be denied by any of the parties of the transaction.
  - Confidentiality(기밀성): data and services are protected from unauthorized access.
  - Integrity(무결성) : system data and services are delivered as intended.
  - Assurance: the parties of a transaction are who they purport to be.
  - Availability: the system will be available for legitimate use.
  - Auditing(감사): tracking activities within the system at levels sufficient enough for reconstruction of events.

보안성 설계전략 (2/5)

- **General Scenario Considerations**:

Source	Human or another system which may have been previously identified (either correctly or incorrectly) or may be currently unknown. A human attacker may be from outside the organization or from inside the organization
Stimulus	Unauthorized attempt is made to display data, change or delete data, access system services, change the system's behavior, or reduce availability
Artifact	**System services, data within the system, a component or resources of the system, data produced or consumed by the system**
Environment	The system is either online or offline; connected to or disconnected from a network; either behind a firewall or open to a network; fully operational, partially operational, or not operational.

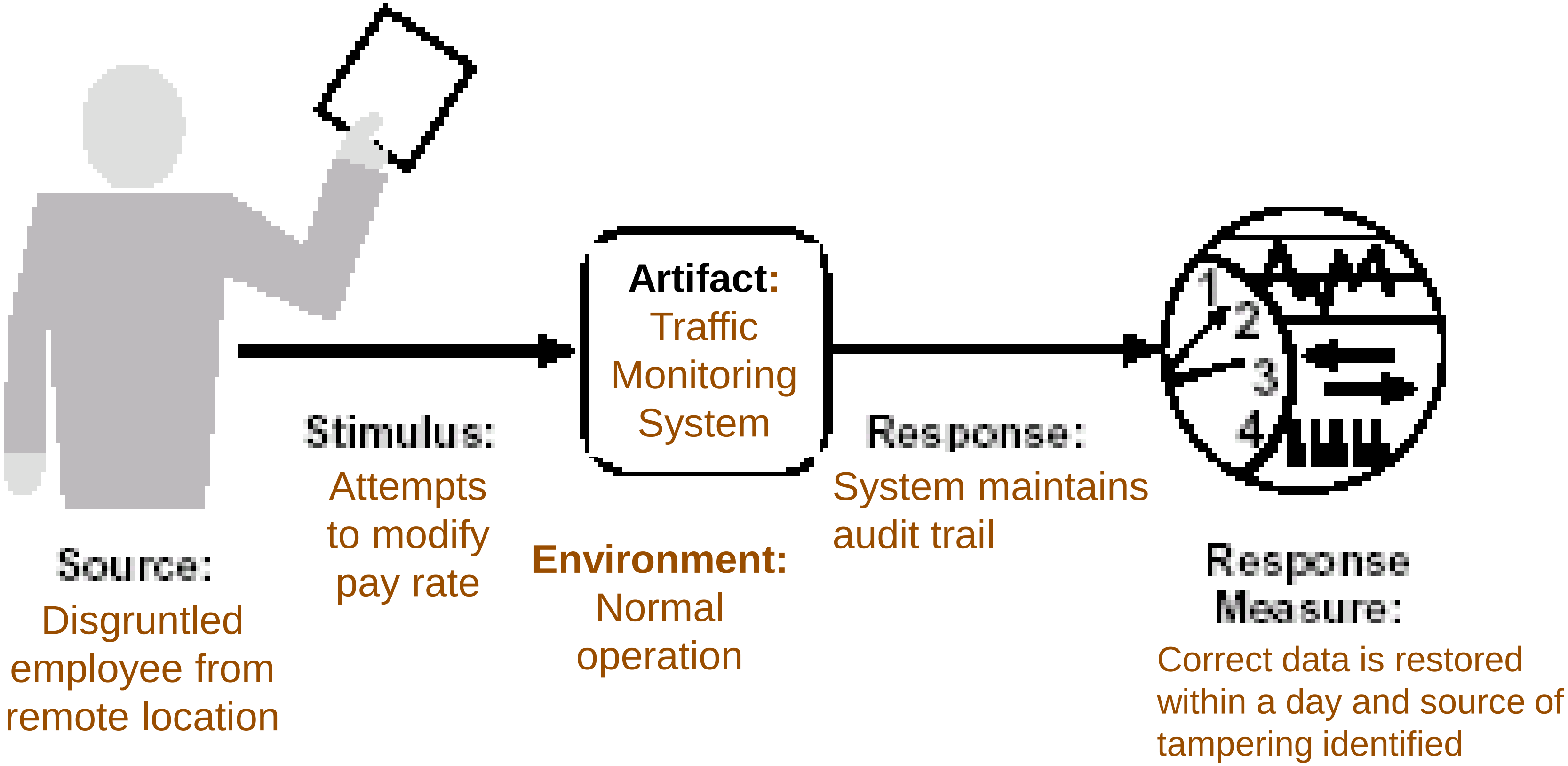
보안성 설계전략 (3/5)

- **General Scenario Considerations (Cont.**.):

Response	Transactions are carried out in a fashion such that - Data or services are protected from authorized access - Data or services are not being manipulated without authorization - Parties to a transaction are identified with assurance - The parties to the transaction cannot repudiate their involvements The data, resources, and system services will be available for legitimate use. The system tracks activities within it by - Recording access or modification - Recording attempts to access data, resources, or services - Notifying appropriate entities (people or systems) when an apparent attack is occurring
Response Measure	One or more of the following: - How much of a system is compromised when a particular component or data value is compromised - How much time passed before an attack was detected - How many attacks were resisted - How long does it take to recover from a successful attack - How much data is vulnerable to a particular attack

보안성 설계전략 (4/5)

- 보안성 품질속성 시나리오 예 - *A correctly identified individual tries to modify system data from an external site. The system maintains an audit trail and the correct data is restored within one day.*



Source:
Disgruntled employee from remote location

Stimulus:
Attempts to modify pay rate

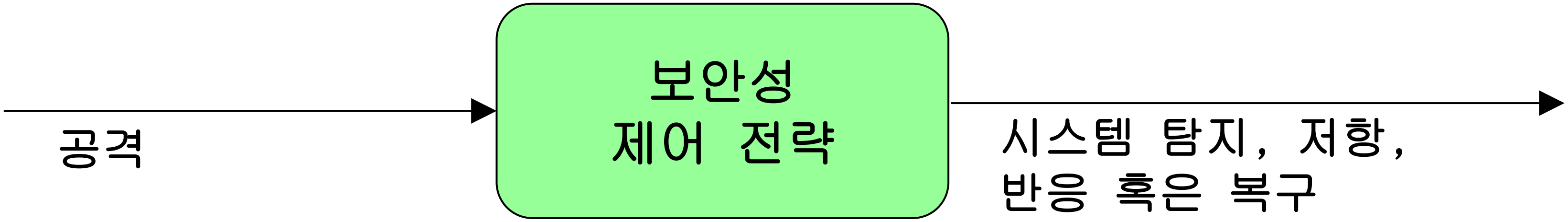
Artifact:
Traffic Monitoring System

Environment:
Normal operation

Response:
System maintains audit trail

Response Measure:
Correct data is restored within a day and source of tampering identified

보안성 설계전략 (5/5)



공격

보안성
제어 전략

시스템 탐지, 저항,
반응 혹은 복구

그림 9-2.보안성 전략의 목표 (출처:[Bass 13]p.151)

공격탐지	공격저항	공격에 반응	공격으로부터 복구	
			Audit Trail 유지	**Restore**
침입탐지 Service Denial 탐지 메시지 무결성 검증 메시지 지연 탐지	Actor들을 식별 Actor들을 인증(authentication) Actor들에 대한 권한제어 (authorization) 접근 제한 노출 제한 데이터 암호화 데이터 entity들을 분리 기본 설정을 변경	접근권한 철회 컴퓨터를 잠금 Actor들에게 통보		*가용성 참조

10.3 품질속성 설계 전략의 정의 절차

- 구체적인 품질속성 설계 전략의 정의 절차 [Bass 13]:
  1. 관련 품질속성을 위한 분석모델로 시작한다.
  2. 그 모델의 파라미터를 식별한다.
  3. 그 모델의 파라미터를 조정하기 위한 아키텍처적 기법을 식별한다.

10.4 전략 적용 사례

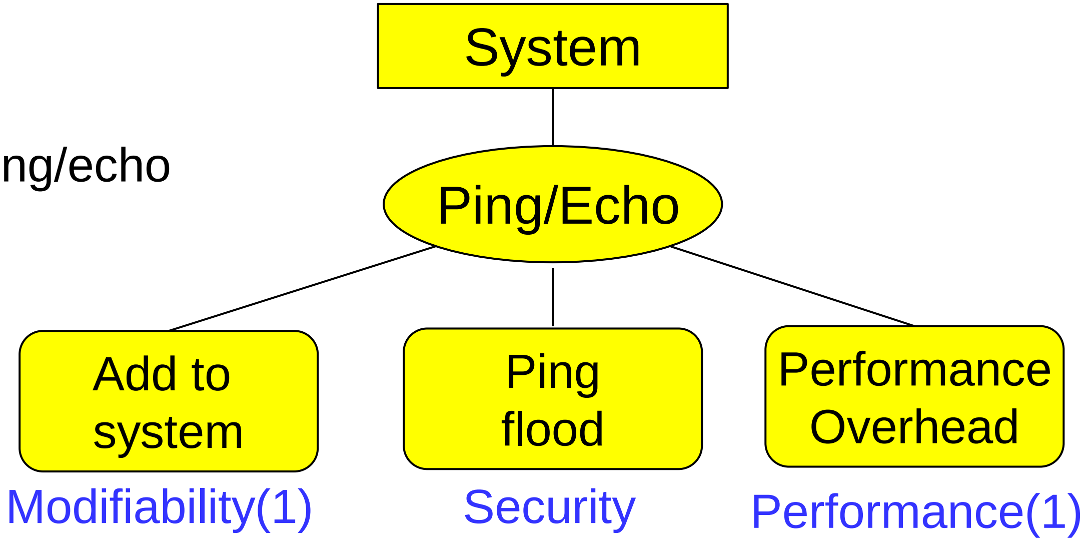
- [Bass 13]pp.242-247:

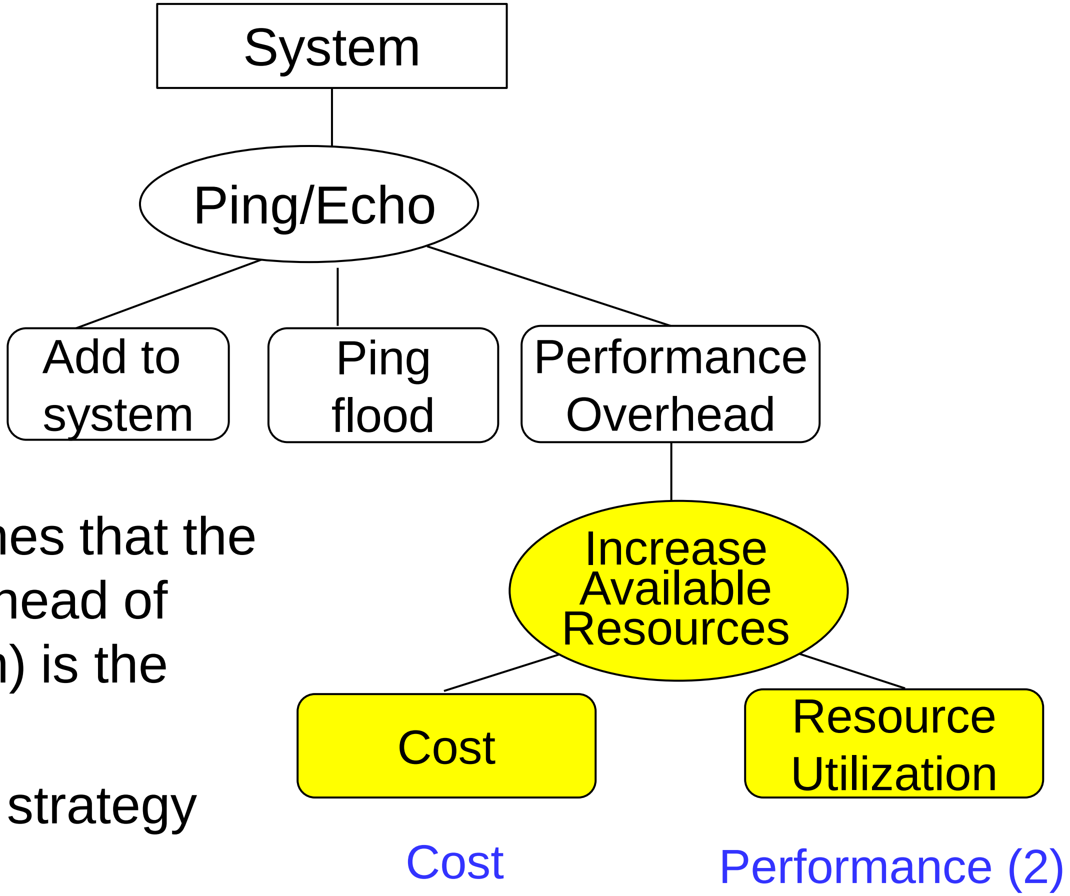
  "Consider a system that needs to detect faults in its components."

=> Decided to use "ping/echo" strategy

=> However, need to consider:

- **Security**: How to prevent a ping flood attack?
- **Performance(1)**: How to ensure that the performance overhead of ping/echo is small?
- **Modifiability(1)**: How to add ping/echo to existing architecture?



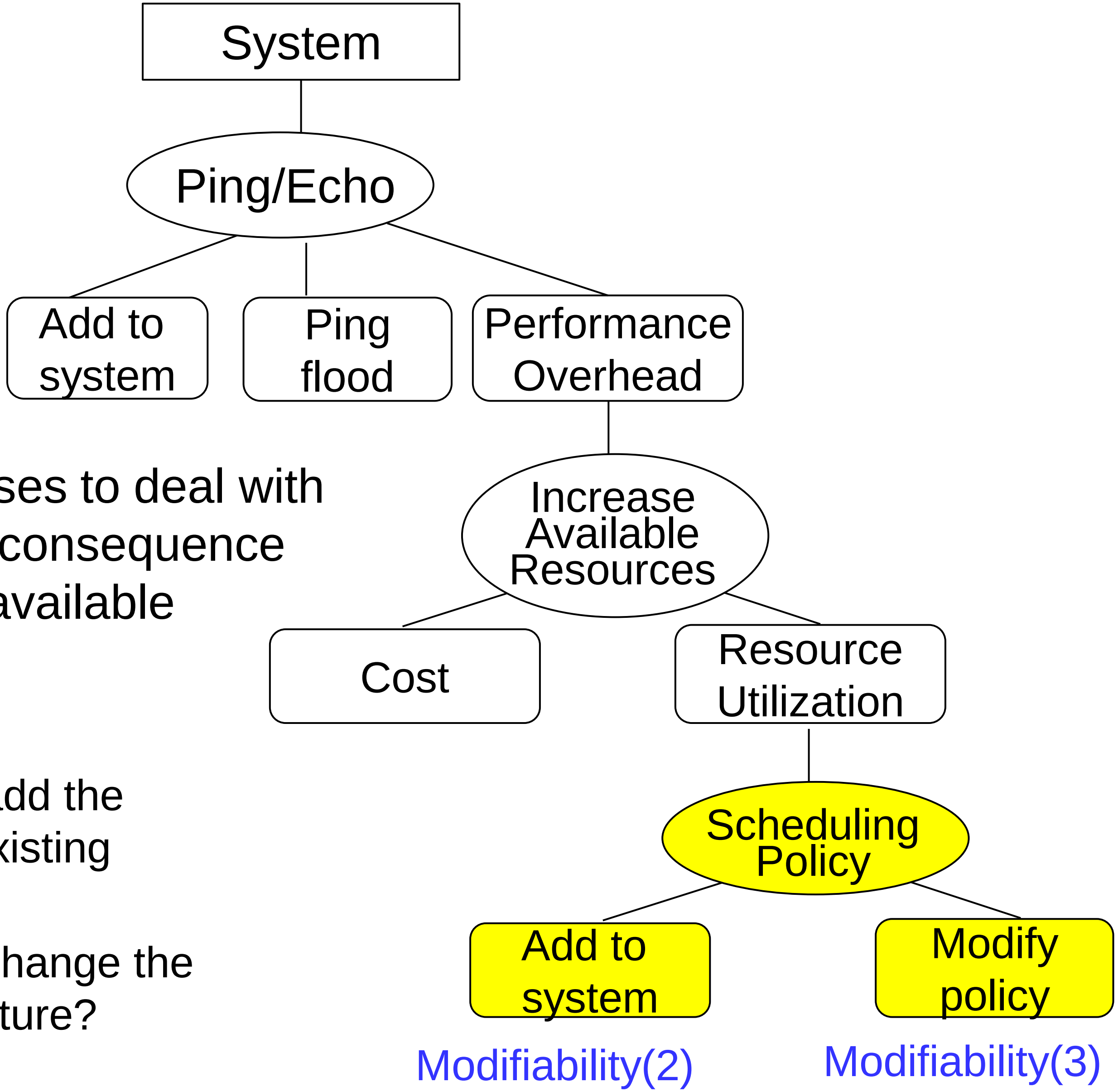


- Suppose the architect determines that the performance tradeoff (the overhead of adding ping/echo to the system) is the most severe.

=> "Increase available resources" strategy

=> Need to consider:

- **Cost**: Increased resources cost more.
- **Performance(2)**: How to utilize the increased resources efficiently?



- Now the architect chooses to deal with the resource utilization consequence of employing increase available resources.

=> Need to consider:

- **Modifiability(2)**: How to add the scheduling policy to the existing architecture?.
- **Modifiability(3)**: How to change the scheduling policy in the future?

- Next the architect chooses to deal with the modifiability (2)

=> "Use an intermediary"

strategy

=> Need to consider:

- **Modifiability(4)**: How to ensure that all communication passes through the intermediary?

=> Need to consider:

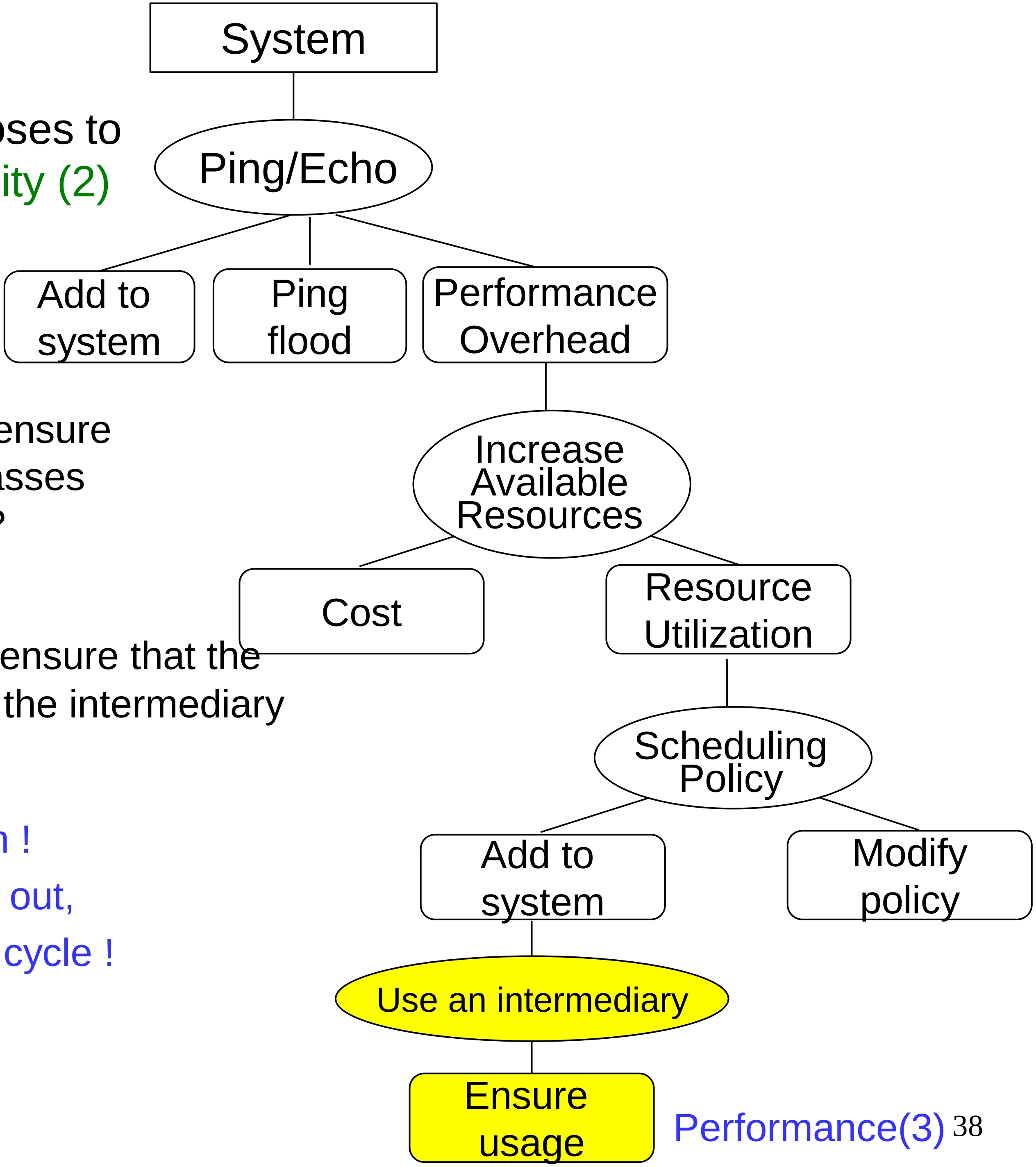
- **Performance(3)**: How to ensure that the performance overhead of the intermediary s is not excessive?

Back to performance problem !

Unless we can "see" the way out,

we will remain trapped in the cycle !

☛ No silver bullet !



Performance(3)

Questions?