Quality Attribute Scenarios

Software Architecture
3rd Year – Semester 1
Lecture 12

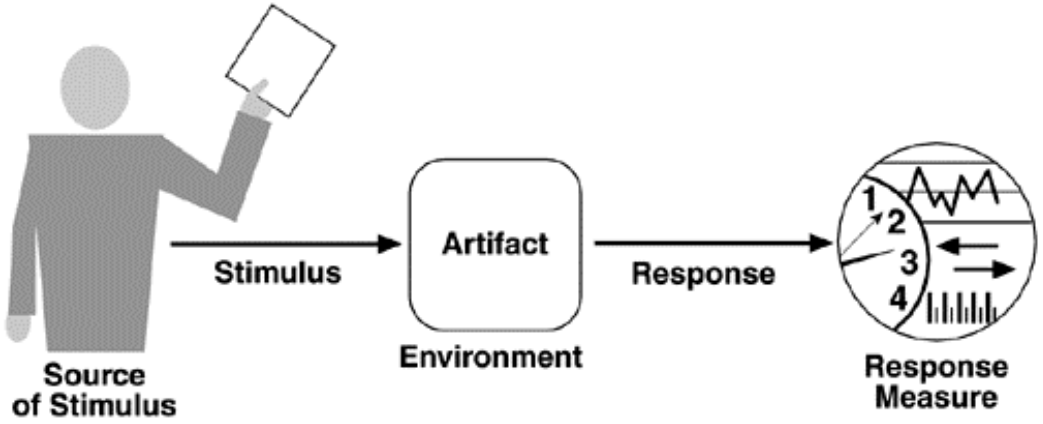
Quality Attribute Scenarios (QAS)

- A universal and formal way to express Quality Attributes.
- The goal of a QAS is to capture unambiguous and testable quality requirements in the same way as use case scenarios do for functional requirements.

General Vs. Concrete Quality Attribute Scenarios

- A general scenario is system independent and can, potentially, pertain to any system.
- A concrete scenario is specific to the particular system under consideration.
- Concrete scenarios are needed to make the quality requirements operational.
- A collection of concrete scenarios can be used as the quality attribute requirements for a system.

Template for QAS



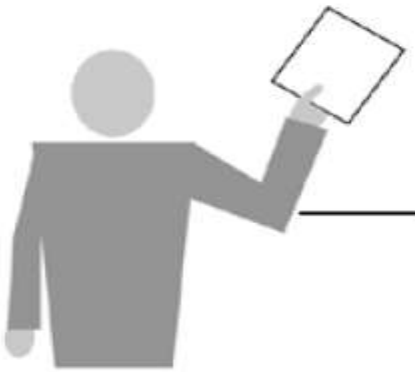
1. **SOURCE:** identifies the originator of the event or action: it can be a user or another system *[who]*
2. **STIMULUS:** describes the action or the external event that arrives at the system *[action]*
3. **ENVIRONMENT:** describes the external circumstances under which the quality requirement needs to be met *[when]*
4. **ARTIFACT:** indicates the part of the system to which the quality requirement applies *[what]*
5. **RESPONSE:** tells us how the system reacts to the stimulus *[result]*
6. **RESPONSE MEASURE:** provides metrics and quantifies the quality attribute *[measurement]*

Availability (QAS)

- Concerned with system failure and it's consequences
- Faults and failures
  - Using the wrong algorithm for computation
  - Miscalculation / incorrect output
- Concerns on failure
  - Frequency
  - Results
  - Non-operative time
  - Prevention
  - Notifications
- The availability of a system is the probability that it will be operational when it is needed

$$\alpha = \frac{\text{mean time to failure}}{\text{mean time to failure} + \text{mean time to repair}}$$

Availability General Scenario

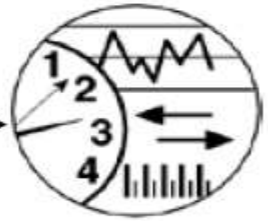


Artifact

System's processors

Communication channels

Persistent storage



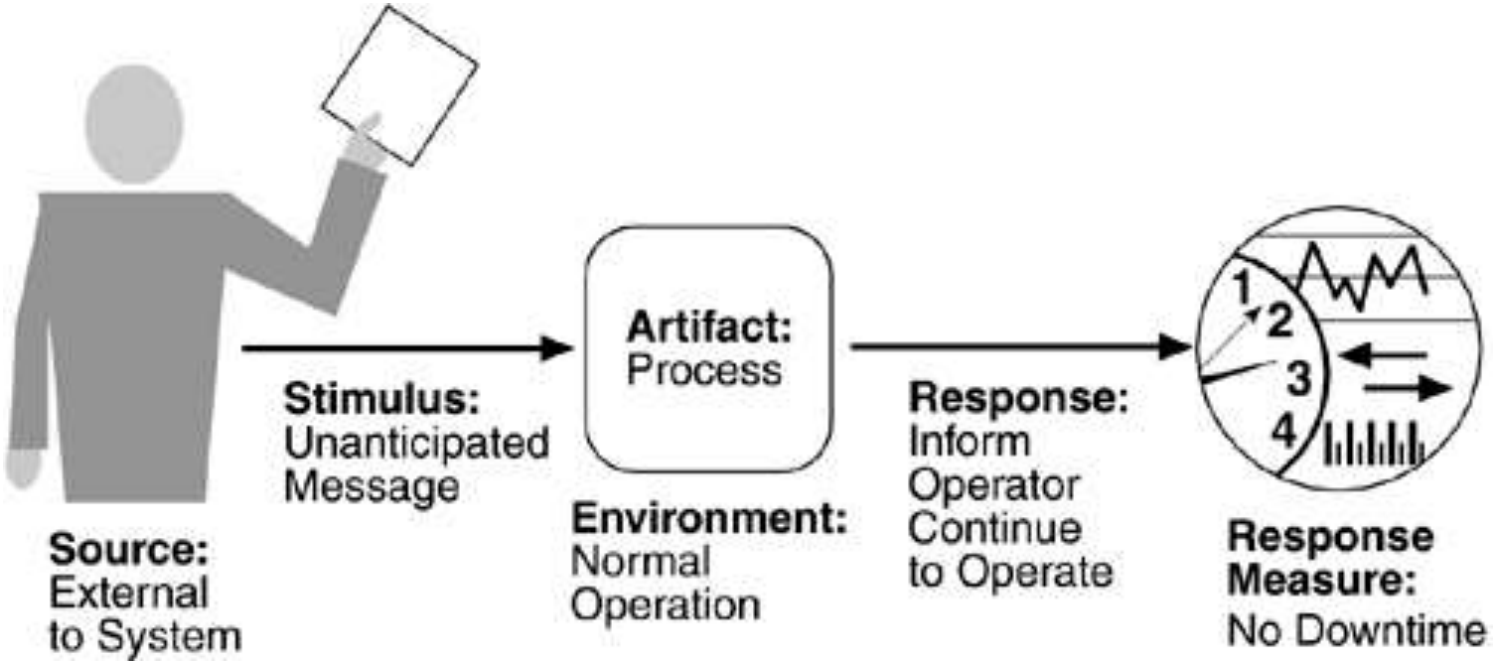
Source	Stimulus	Environment	Response	Measure
Internal to system	Crash	Normal operation	Prevent the failure	Time interval available
External to system	Omission	Startup	Log the failure	Availability %
	Timing	Shutdown	Notify users / operators	Detection time
	No response	Repair mode	Disable source of failure	Repair time
	Incorrect response	Degraded (failsafe) mode	Temporarily unavailable	Degraded mode time interval
		Overloaded operation	Continue (normal / degraded)	Unavailability time interval

Availability General Scenario

Portion of Scenario	Possible Values
Source	Internal to the system; external to the system
Stimulus	Fault: omission, crash, timing, response
Artifact	System's processors, communication channels, persistent storage, processes
Environment	Normal operation;
	degraded mode (i.e., fewer features, a fall back solution)
Response	System should detect event and do one or more of the following:
	record it
	notify appropriate parties, including the user and other systems
	disable sources of events that cause fault or failure according to defined rules
	be unavailable for a prespecified interval, where interval depends on criticality of system
	continue to operate in normal or degraded mode
Response Measure	Time interval when the system must be available
	Availability time
	Time interval in which system can be in degraded mode
	Repair time

Availability Concrete Scenario

E.g. An unanticipated external message is received by a process during normal operation. The process informs the operator of the receipt of the message and continues to operate with no downtime.



Exercise: Availability Concrete Scenario

Q: Write a Concrete Quality Attribute Scenario for Word Application.

A: Kill Signal is received from the Windows OS to Ms. Word Application during process Not Responding state and the application saves unsaved work in a temp file and process terminates without any data loss.

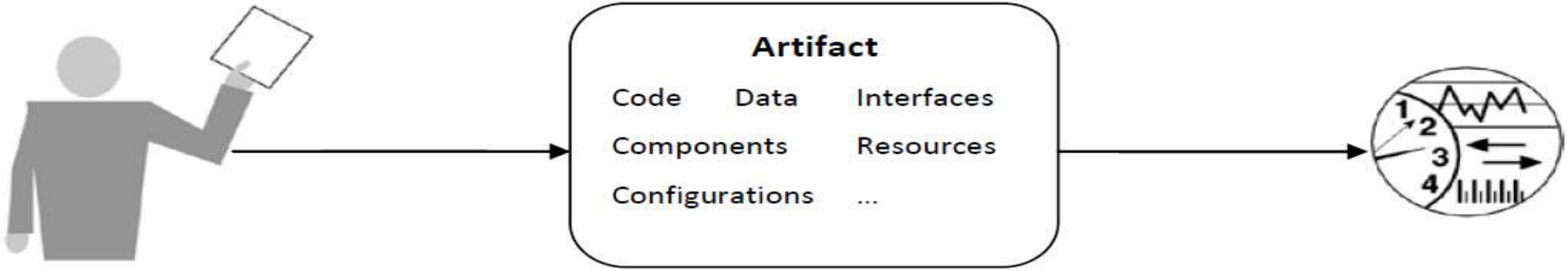
Identify:

- Source
- Stimulus
- Artifact
- Environment
- Response
- Response Measure

Modifiability (QAS)

- Modifiability is about the cost of change.
- It brings up two concerns:
  - What can change (the artifact)?
    - Functions, Platform, Environnent, Protocol, Qualities, Capacity
  - When is the change made and who makes it (the environment)?
    - A developer, an end user, or a system administrator.
    - At Implementation, Compilation, Build, Configuration, Execution

Modifiability General Scenario



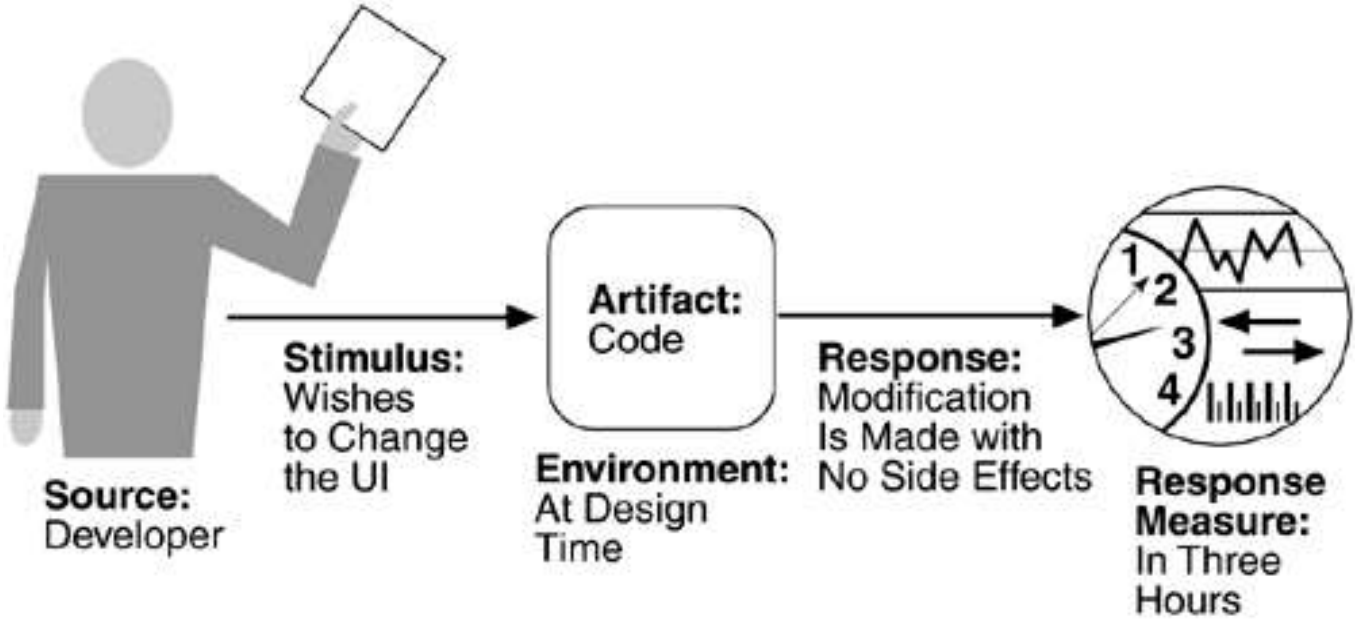
Source	Stimulus	Environment	Response	Measure
End-user	Add / delete / modify functionality, quality attribute, capacity or technology	Runtime	Make modification	Cost in effort
Developer		Compile time	Test modification	Cost in money
System-administrator		Build time	Deploy modification	Cost in time
		Initiation time		Cost in number, size, complexity of affected artifacts
		Design time		Extent affects other system functions or qualities
				New defects introduced

Modifiability General Scenario

Portion of Scenario	Possible Values
Source	End user, developer, system administrator
Stimulus	Wishes to add/delete/modify/vary functionality, quality attribute, capacity
Artifact	System user interface, platform, environment; system that interoperates with target system
Environment	At runtime, compile time, build time, design time
Response	Locates places in architecture to be modified; makes modification without affecting other functionality; tests modification; deploys modification
Response Measure	Cost in terms of number of elements affected, effort, money; extent to which this affects other functions or quality attributes

Modifiability Concrete Scenario

E.g. A developer wishes to change the user interface to make a screen's background color blue. This change will be made to the code at design time. It will take less than three hours to make and test the change and no side effect changes will occur in the behavior.



Exercise: Modifiability Concrete Scenario

Q: Write a Concrete Quality Attribute Scenario for updating database Password on a 3 Tier application.

A: System Administrator wishes to change the password of the database configuration file on the Data Tier at the System Maintenance Time; the activity takes 2 minutes and the application is able to connect to the Database without any issues.

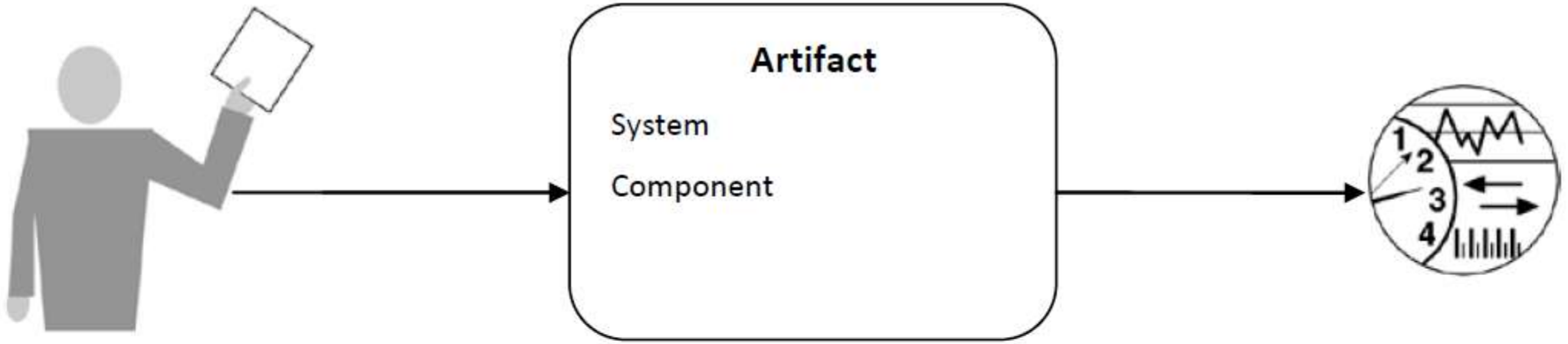
Identify:

- Source
- Stimulus
- Artifact
- Environment
- Response
- Response Measure

Performance (QAS)

- Performance is about timing. Events (interrupts, messages, requests from users, or the passage of time) occur, and the system must respond to them.
- There are a variety of characterizations of event arrival and the response but basically performance is concerned with how long it takes the system to respond when an event occurs

Performance General Scenario



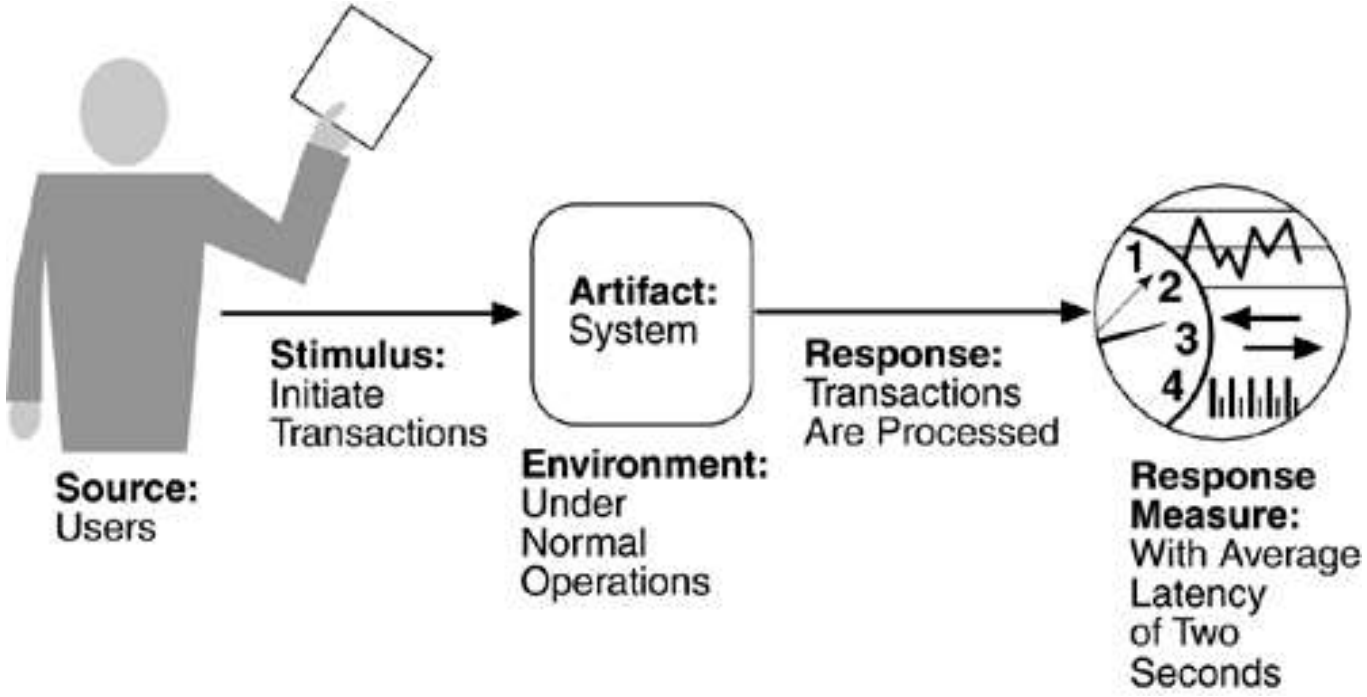
Source	Stimulus	Environment	Response	Measure
Internal to the system	Periodic events	Normal mode	Process events	Latency
External to the system	Sporadic events	Overload mode	Change level of service	Deadline
	Bursty events	Reduced capacity mode		Throughput
	Stochastic events	Emergency mode		Jitter
		Peak mode		Miss rate
				Data loss

Performance General Scenario

Portion of Scenario	Possible Values
Source	One of a number of independent sources, possibly from within system
Stimulus	Periodic events arrive; sporadic events arrive; stochastic events arrive
Artifact	System
Environment	Normal mode; overload mode
Response	Processes stimuli; changes level of service
Response Measure	Latency, deadline, throughput, jitter, miss rate, data loss

Performance Concrete Scenario

E.g. Users initiate 1,000 transactions per minute randomly under normal operations, and these transactions are processed with an average latency of two seconds.



Exercise: Performance Concrete Scenario

Q: Write a Concrete Quality Attribute Scenario for A banking application for a Weekly Transaction Report.

A: The Finance Analyst schedules weekly report of the Banking Application during Normal Operational Time (8am-5pm), the process starts to execute at the Off-Peak Time (between 10pm-2am) and the Report Excel File generates successfully within 30 minutes.

Identify:

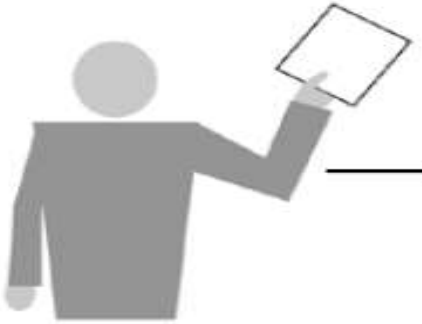
- Source
- Stimulus
- Artifact
- Environment
- Response
- Response Measure

Max Latency = 18 hours (8am –> 2am)
Max Deadline = latency + 30 mins

Security (QAS)

- Security is a measure of the system's ability to resist unauthorized usage while still providing its services to legitimate users.
- An attempt to breach security is called an attack and have many forms;
  - Unauthorized attempt to access data
  - Modify data
  - Intended to deny services to legitimate users.
- Characterization
  - Nonrepudiation : Transaction cannot be denied by any of the parties
  - Confidentiality : Data or services are protected from unauthorized access.
  - Integrity : Data or services are being delivered as intended.
  - Assurance or authenticity : The parties to a transaction are who they purport to be
  - Availability (no denial of service) : The system will be available for legitimate use
  - Auditing : The system tracks activities

Security General Scenario



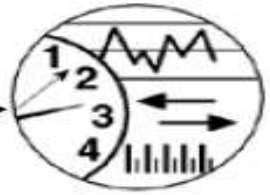
Artifact

System services

Data within the system

Component / resource of the system

Data produced / consumed by the system



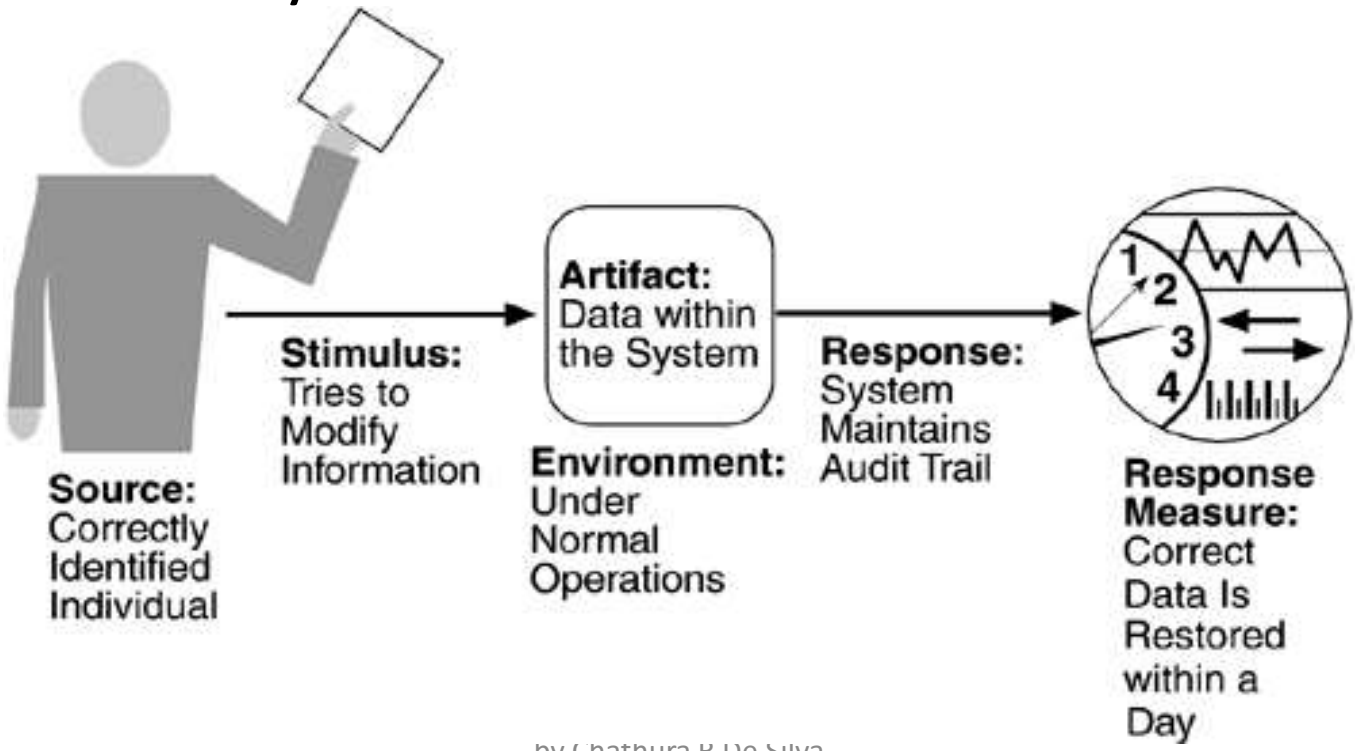
Source	Stimulus	Environment	Response	Measure
Identified user	Attempt to display data	Online or Offline	User Authentication Identification	Time/Effort/ Resources
Unknown user	Attempt to modify data	Connected or Disconnected	Allow/Blocks Access	Probability of Success
Hacker from outside the organization	Attempt to delete data	Firewalled or Open	Grant/Withdraw Permission	Probability of Detection
Hacker from inside the organization	Access system services		Data Readability	Percentage of Accesibility
	Change system's behavior			
	Reduce availability			

Security General Scenario

Portion of Scenario	Possible Values
Source	Individual or system that is correctly identified, identified incorrectly, of unknown identity who is internal/external, authorized/not authorized with access to limited resources, vast resources
Stimulus	Tries to display data, change/delete data, access system services, reduce availability to system services
Artifact	System services; data within system
Environment	Either online or offline, connected or disconnected, firewalled or open
Response	Authenticates user; hides identity of the user; blocks access to data and/or services; allows access to data and/or services; grants or withdraws permission to access data and/or services; records access/modifications or attempts to access/modify data/services by identity; stores data in an unreadable format; recognizes an unexplainable high demand for services, and informs a user or another system, and restricts availability of services
Response Measure	Time/effort/resources required to circumvent security measures with probability of success; probability of detecting attack; probability of identifying individual responsible for attack or access/modification of data and/or services; percentage of services still available under denial-of-services attack; restore data/services; extent to which data/services damaged and/or legitimate access denied

Security Concrete Scenario

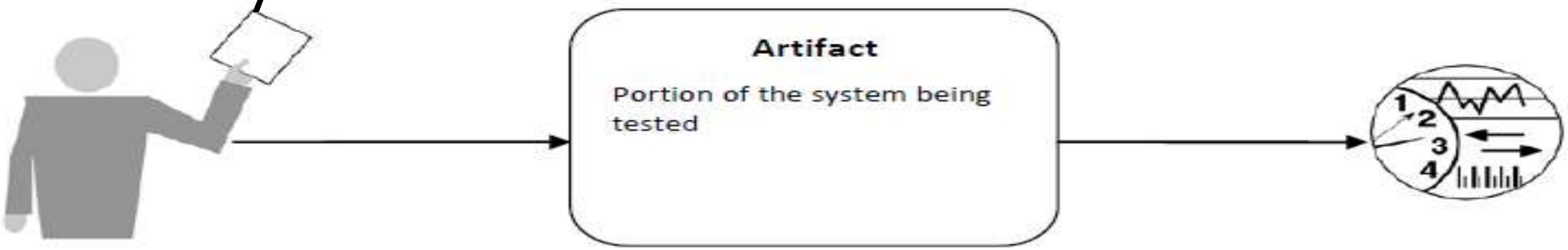
E.g. A correctly identified individual tries to modify system data from an external site; system maintains an audit trail and the correct data is restored within one day.



Testability (QAS)

- Software testability refers to the ease with which software can be made to demonstrate its faults through (typically execution-based) testing.
  - At least 40% of the cost of developing well-engineered systems is taken up by testing.
- For a system to be properly testable, it must be possible to control each component's internal state and inputs and then to observe its outputs

Testability General Scenario



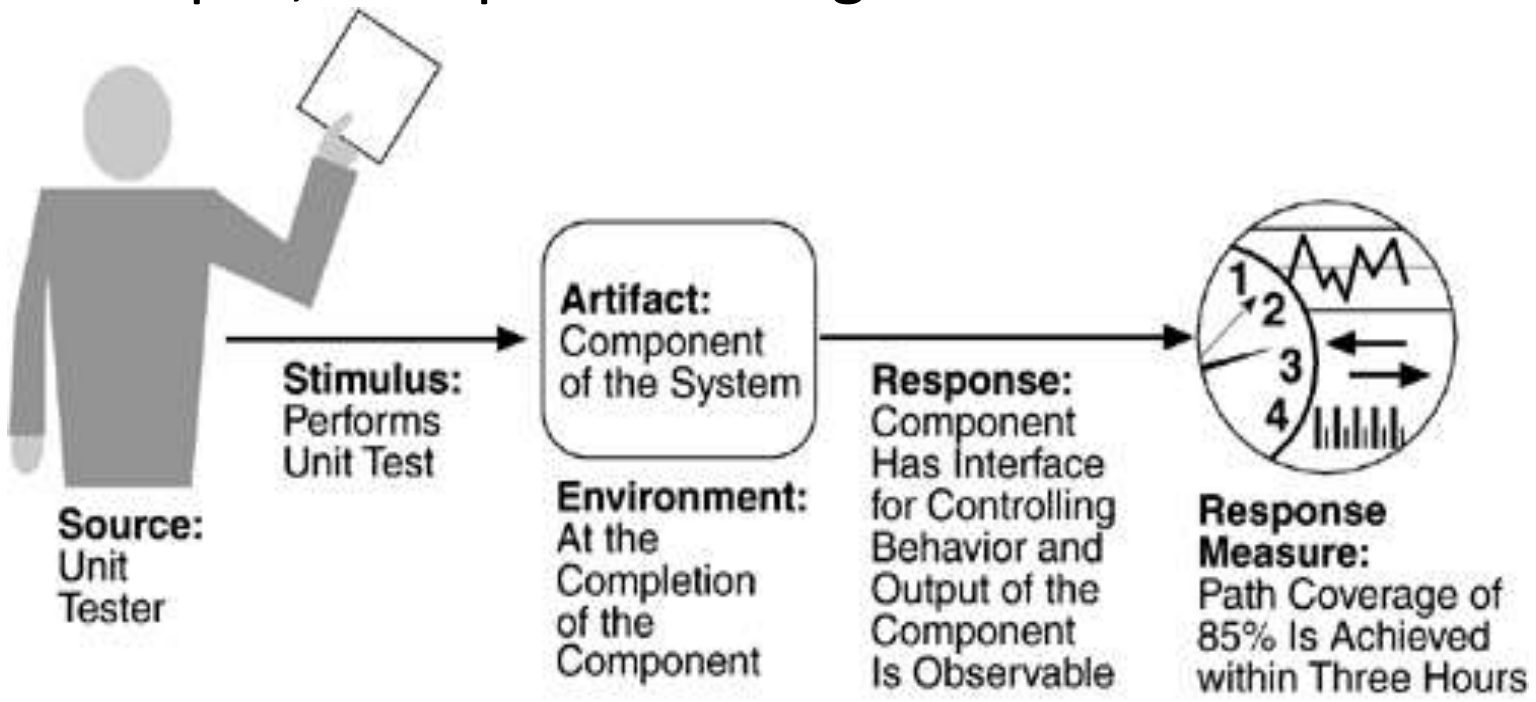
Source	Stimulus	Environment	Response	Measure
Unit tester	Execution of tests due to completion of code increment	Design time	Execute test suite & capture results	Effort to find fault
Integration tester		Development time	Capture cause of fault	Effort to achieve coverage %
System tester		Compile time	Control & monitor state of the system	Probability of fault being revealed by next test
Acceptance tester		Integration time		Time to perform tests
End user		Deployment time		Effort to detect faults
Automated testing tools		Run time		Length of longest dependency chain
				Time to prepare test environment
				Reduction in risk exposure

Testability General Scenario

Portion of Scenario	Possible Values
Source	Unit developer; Increment integrator; System verifier; Client acceptance tester; System user
Stimulus	Analysis, architecture, design, class, subsystem integration completed; system delivered
Artifact	Piece of design, piece of code, complete application
Environment	At design time, at development time, at compile time, at deployment time
Response	Provides access to state values; provides computed values; prepares test environment
Response Measure	Percentage of the statements executed; Probability of failure if fault exists; Time to perform tests; Length of longest dependency chain in a test Length of time to prepare test environment

Testability Concrete Scenario

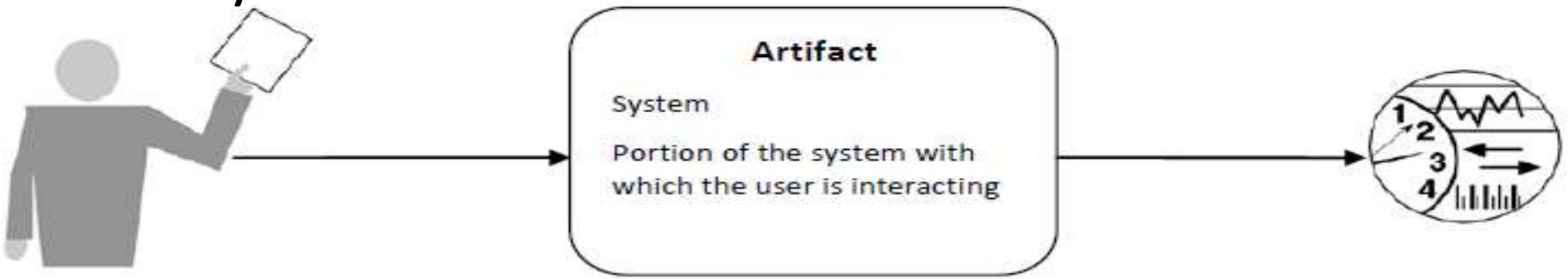
E.g. A unit tester performs a unit test on a completed system component that provides an interface for controlling its behavior and observing its output; 85% path coverage is achieved within three hours.



Usability (QAS)

- How easy it is to learn the features of the system
- How efficiently the user can use the system
- How well the system handles user errors
- How well the system adapts to user needs
- To what degree the system gives the user confidence in the correctness of its actions.

Usability General Scenario



Source

End user (possibly special role)

Stimulus

Use the system efficiently

Learn to use the system

Minimize impact of errors

Adapt the system

Configure the system

Environment

Runtime

Configuration time

Response

Provide features needed

Anticipate the user's needs

Measure

Task time

Number of errors

Number of tasks accomplished

User satisfaction

Gain of user knowledge

Ratio of successful operations to total operations

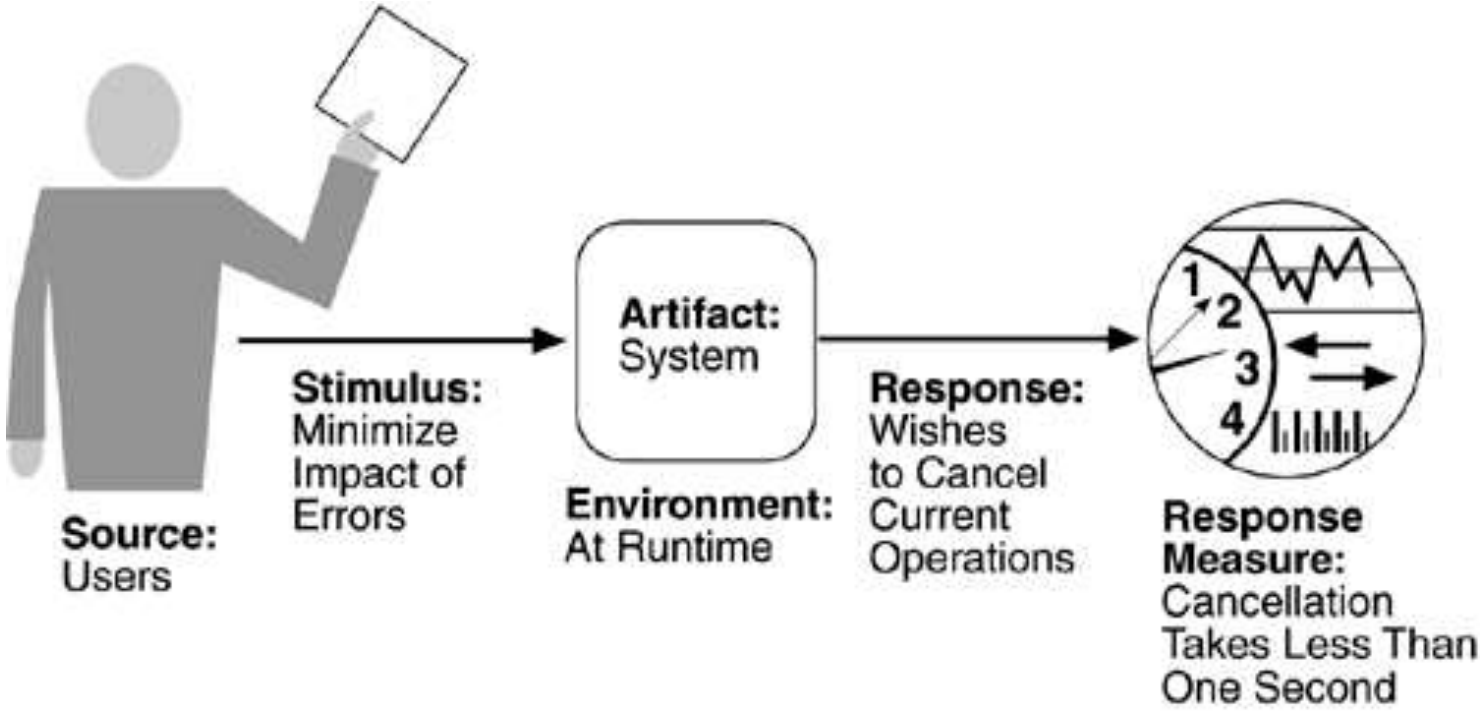
Amount of time / data lost when error occurs

Usability General Scenario

Portion of Scenario	Possible Values
Source	End user is always the source (can be broken down to user roles/actors)
Stimulus	Wants to learn system features; use system efficiently; minimize impact of errors; adapt system; feel comfortable
Artifact	System (or a part of the system the user is interacting)
Environment	At runtime or configure time
Response	System provides one or more of: • To support learn system features • To support use system efficiently • To minimize impact of errors • To adapt system: customizability; internationalization • To feel comfortable: display system state; work at the user's pace
Response Measure	Task time, number of errors, number of problems solved, user satisfaction, gain of user knowledge, ratio of successful operations to total operations, amount of time/data lost

Usability Concrete Scenario

E.g. A user, wanting to minimize the impact of an error, wishes to cancel a system operation at runtime; cancellation takes place in less than one second.



Exercise (offline)

- Refer a few existing software systems (e.g. your Group Case Study) and identify 2-3 quality attribute scenarios for the Main Quality Attribute(s) Under consideration
- Check with the actual Software System documentation / agreements / guides if the above identified Quality Attribute response measures are stated
- Propose new Quality Attribute Scenarios for your system under considerations

Tactics

- An architectural tactic is a means of satisfying a quality attribute response measure by manipulating some aspect of a quality attribute model through architectural decisions

NEXT LECTURE:

Ways to improve Quality Attributes

Tactics Framework

References

- http://www.ece.ubc.ca/~matei/EECE417/BASS/ch04lev1sec4.html
- http://etutorials.org/Programming/Software+architecture+in+practice,+second+edition/Part+Two+Creating+an+Architecture/Chapter+4.+Understanding+Quality+Attributes/4.4+Quality+Attribute+Scenarios+in+Practice/