Quality Attributes
CSSE 574: Session 8, Part 7 (!)

Steve Chenoweth
Phone: Office (812) 877-8974, Cell (937) 657-3885
Email: chenowet@rose-hulman.edu
This discussion

- Quality attributes –
  - Bass’s core concept
  - This is Ch 5-12 in Bass
Acknowledgements

Some of the material in these slides is taken from *Software Architecture in Practice, 2nd edition* by Bass, Clements and Kazman. Ch 4, and *3rd edition*, Ch 5-12.

Due as usual to prior authors of this course – Steve, Mark, Lisa:

Mark Ardis is now at RIT, see [http://www.se.rit.edu/se/people/facDetail.maml?t_Link=5&t_RenderAs=edit](http://www.se.rit.edu/se/people/facDetail.maml?t_Link=5&t_RenderAs=edit).

Lisa Kaczmarczyk is now Chief Technology Officer of Sixth College at UCSD, see [http://www.ucsd.edu/directory/faculty_staff?entry=kaczmarczyk](http://www.ucsd.edu/directory/faculty_staff?entry=kaczmarczyk).
Outline

- Business and architectural qualities which lead to system qualities we want
- System qualities (Bass’s list, which we’ll use) –
  - Availability
  - Interoperability
  - Performance
  - Security
  - Modifiability
  - Testability
  - Usability
Business Qualities

- Time to market – “Time”
- Cost and benefit – “Economy”
- Projected lifetime – “Form”
- Target market – “Function”
- Rollout schedule – “Time”
- Integration with legacy – “Time”

We need these business qualities to persist, so your software needs Architectural Qualities…
Architectural Qualities

- Conceptual integrity
- Correctness and completeness
- Buildability

- Very close ties to “Form” part of F/F/E/T model in problem statement template
System Qualities

- What system qualities are needed to achieve those architectural qualities?
  - Availability
  - Interoperability
  - Performance
  - Security
  - Modifiability
  - Testability
  - Usability

- And we did put all these under “Form” in the Problem Statement.

- Same ones asked for in the suppl spec template – Let’s discuss the “scenarios” for these…
Quality Scenarios - General

Source: Internal, External

Stimulus: (Fault) Omission, Crash, Timing, Response

Artifact: Process, Storage, Processor, Communication

Environment: Normal, Degraded Operation

Response: Record, Notify, Disable, Continue (Normal/Degraded) Be Unavailable

Response Measure: Repair, Time, Availability, Available/ Degraded Time Interval
Quality Scenarios - Concrete

Source: External to System

Stimulus: Unanticipated Message

Artifact: Process

Environment: Normal Operation

Response: Inform Operator Continue to Operate

Response Measure: No Downtime
Quality Scenarios - Table

- Source: who?
- Stimulus: what?
- Artifact: where?
- Environment: when?
- Response: which?
- Measure: how?
More about each…

- Bass’s seven quality attributes
- Scenarios for each, with examples
- This info is the main appendix to your suppl spec template.
Availability

A runtime attribute:
- Failures and faults
- Mean time to failure, repair
- Downtime

\[ A = \frac{E[\text{Uptime}]}{E[\text{Uptime}] + E[\text{Downtime}]} \]

### Availability Table

- **Source:** internal, external
- **Stimulus:** type of fault
- **Artifact:** processors, channels, storage
- **Environment:** normal, degraded
- **Response:** logging, notification, switching to backup, restart, shutdown
- **Response Measure:** availability, repair time, required uptime
Sample availability scenario

- **Source:** External to the system
- **Stimulus:** Unanticipated message
- **Artifact:** Process
- **Environment:** Normal operation
- **Response:** Inform operator continue to operate
- **Response Measure:** No downtime
Interoperability

- Need is discovered either ahead of runtime or at runtime!
- You either offer a service or use one.
- Key aspects are:
  - Discovery
  - Disposition - Handling the response
  - Managing the interfaces

Interoperability Table

- **Source:** A system initiates a request to interoperate with another system.
- **Stimulus:** A request to exchange information among system(s).
- **Artifact:** The systems that wishes to interoperate.
- **Environment:** The systems that wishes to interoperate are discovered at runtime or are known prior to runtime.
- **Response:** The request to interoperate results in the exchange of information. The information is understood by the receiving party both syntactically and semantically. Alternatively, the request is rejected and appropriate entities are notified. In either case, the request may be logged.
- **Response Measure:** The percentage of information exchanges correctly processed or the percentage of information exchanges correctly rejected.
Sample Interoperability Scenario

- **Source:** Our vehicle information system
- **Stimulus:** Current location sent
- **Artifact:** Traffic monitoring system
- **Environment:** Systems known prior to runtime
- **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
Performance

A runtime attribute:

- Event arrival patterns
  - periodic
  - stochastic
  - sporadic

- Event servicing
  - latency
  - jitter
  - throughput
Performance

A typical analysis (M/M/1, from queueing theory):

Performance Table

- **Source**: external, internal
- **Stimulus**: event arrival pattern
- **Artifact**: system services
- **Environment**: normal, overload
- **Response**: change in mode?
- **Measure**: latency, deadline, throughput, jitter, miss rate, data loss
Sample performance scenario

- **Source:** Users
- **Stimulus:** Initiate transactions
- **Artifact:** System
- **Environment:** Under normal operations
- **Response:** Transactions are processed
- **Response Measure:** With average latency of two seconds
Security

A runtime attribute:
- Attack or threat
- Confidentiality
- Integrity
- Assurance
- Availability

Security Table

- **Source**: user/system, identified?
- **Stimulus**: display info, change info, access services, deny services
- **Artifact**: services, data
- **Environment**: online/offline, connected?
- **Response**: logging, block access, notification
- **Measure**: time, probability of detection, recovery
Sample security scenario

- **Source:** Correctly identified individual
- **Stimulus:** Tries to modify information
- **Artifact:** Data within the system
- **Environment:** Under normal operations
- **Response:** System maintains audit trail
- **Response Measure:** Correct data is restored within a day
Modifiability

A development-time attribute:

- What can change?
- When is it changed?
- Who changes it?

Modifiability Table

- **Source:** developer, administrator, user
- **Stimulus:** add/delete/modify function or quality
- **Artifact:** UI, platform, environment
- **Environment:** design, compile, build, run
- **Response:** make change and test it
- **Measure:** effort, time, cost
Sample modifiability scenario

- **Source:** Developer
- **Stimulus:** Wishes to change the UI
- **Artifact:** Code
- **Environment:** At design time
- **Response:** Modification is made with no side effects
- **Response Measure:** In 3 hours
Testability

A development-time attribute:
- Probability of fault discovery
- Need to control components
- Need to observe component failure

Right: Ren & Stimpy tell why people ignore system testability. From web site sctest.cse.ucsc.edu/.
Testability Table

- **Source:** developer, tester, user
- **Stimulus:** milestone completed
- **Artifact:** design, code component, system
- **Environment:** design, development, compile, deployment, run
- **Response:** can be controlled and observed
- **Measure:** coverage, probability, time
Sample testability scenario

- **Source:** Unit tester
- **Stimulus:** Performs unit test
- **Artifact:** Component of the system
- **Environment:** At the completion of the component
- **Response:** Component has interface for controlling behavior, and output of the component is observable
- **Response Measure:** Path coverage of 85% is achieved within 3 hours
Usability

A runtime attribute:
- Learning
- Using efficiently
- Minimizing errors
- Adapting to user needs
- Increasing confidence and satisfaction

Below – Usability testers at work. From [http://www.cognetics.com/services/design_services/services_evaluation.html](http://www.cognetics.com/services/design_services/services_evaluation.html).
Usability Table

- **Source:** end user
- **Stimulus:** wish to learn/use/minimize errors/adapt/feel comfortable
- **Artifact:** system
- **Environment:** configuration or runtime
- **Response:** provide ability or anticipate
- **Measure:** task time, number of errors, user satisfaction, efficiency
Sample usability scenario

- **Source:** Users
- **Stimulus:** Minimize impact of errors
- **Artifact:** System
- **Environment:** At runtime
- **Response:** Wishes to cancel current operations
- **Response Measure:** Cancellation takes less than one second