



**Carnegie Mellon
Software Engineering Institute**

Pittsburgh, PA 152 13-3890

Design and Analysis Principles for Software Architecture

Len Bass
Software Engineering Institute


Sponsored by the U.S. Department of Defense
© 2004 by Carnegie Mellon University

page 1


[Next](#)

[Exit Slide Show](#)


[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)

 Carnegie Mellon
Software Engineering Institute


Why build a computer system?



System



Designer



Software Architecture

© 2004 by Carnegie Mellon University page 8

[Previous](#) [Next](#)

[Exit Slide Show](#)

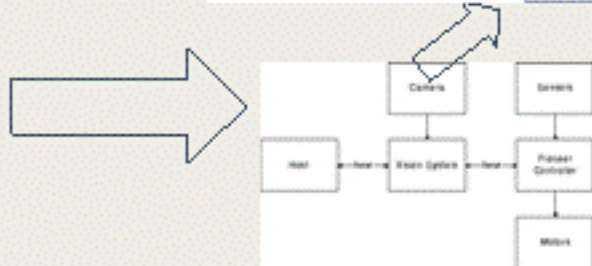
- [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



To Achieve Some Business Goals

Business Goals

- Financial/mission
 - Improve operations
 - Offer new services
- Marketing
 - Extend market share
 - Improve customer satisfaction
- Other
 - Get tenure/degree
 - Make political point



**Software
Architecture**

[Previous](#) [Next](#)

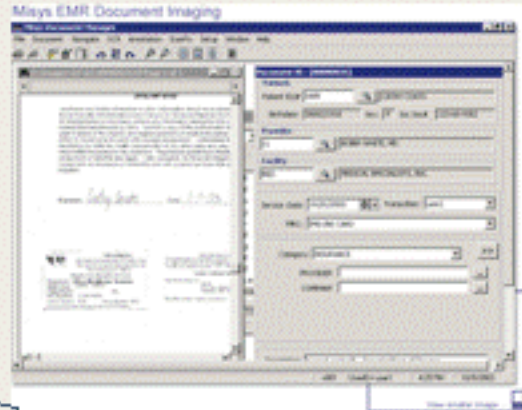
[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

What is the software architecture of a system?



**Software
Architecture**

© 2004 by Carnegie Mellon University

page 7

[Previous](#) [Next](#)

[Exit Slide Show](#)

- [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

The Definition of Software Architecture

The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.

© 2004 by Carnegie Mellon University

page 8

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Implications of Our Definition

Architecture is an abstraction of systems.

- Architecture defines components and how they interact.
- Architecture suppresses purely local information about components; private details are not architectural.

Systems have many structures (views).

- No single structure can be *the* architecture.
- The set of candidate structures is not fixed or prescribed: whatever is useful for analysis, communication, or understanding.

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Business Goals Beget Requirements

Business Goals  **Requirements**

Marketing => strategies

- Time to market
- Portion of market targeted
- Product lines
- Alliances


Functional
Quality
Constraints

© 2004 by Carnegie Mellon University page 12

[Previous](#) [Next](#)


[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Types of Requirements



Requirements

- Constraints – pre-specified design decisions
- Features – what functions add value to the user (e.g. what the system does)
- Quality Attribute– how well the system does by various measures (e.g., how timely, secure, modifiable it is)

Software Architectures

Functions are:

- Features + necessary non user visible computations
- Represented within architecture by responsibilities


© 2004 by Carnegie Mellon University

page 16

[Previous](#) [Next](#)

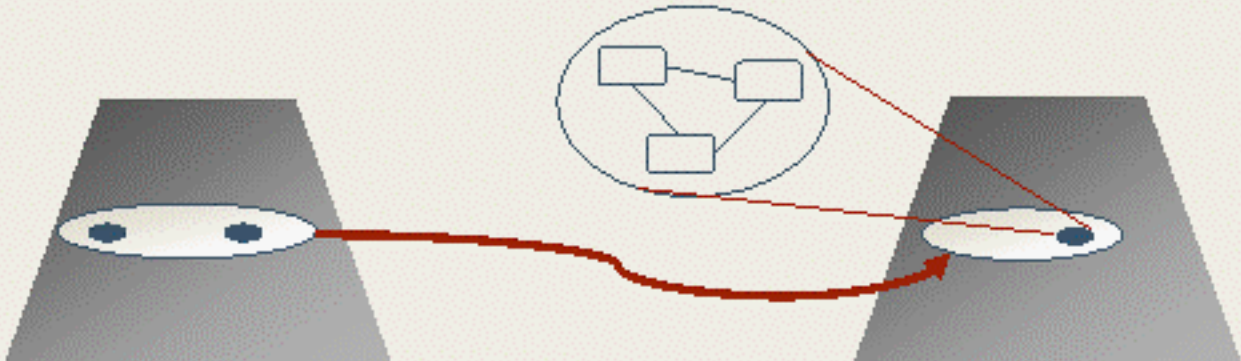
[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Constraints



Requirements

Constraints – pre-specified design decisions (e.g. what the system does)

Software Architectures

Constraints reduce the space of architectures in which to search for a solution

© 2004 by Carnegie Mellon University

page 17

[Previous](#) [Next](#)

[Exit Slide Show](#)

- [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Must live within constraints

Very little software design is "greenfield"

Frameworks, large-grained components are frequently required.

Disciplined design must accommodate constraints.

Designer does not make design decisions to achieve constraints – constraints are given.

Designer makes design decisions to achieve other requirements within given constraints.

© 2004 by Carnegie Mellon University

page 18

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Do functional or quality requirements drive remaining architectural design?

/A/ Quality requirements determine most architectural design decisions – not functional requirements

If the only concern is functionality then a monolithic system would suffice.

However is it quite common to see:

- Redundancy structures for reliability
- Concurrency structures for performance
- Layers for modifiability

© 2004 by Carnegie Mellon University

page 21

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

More evidence

Time to market/cost determines buy/build

- Buying and architecture are inter-twined
- "buildability" is quality attribute related to time to market

Restructuring a system is *usually* done because of quality reasons, not because of functionality

- Improve performance or security
- Improve the *ease* of making modifications, not making modifications, *per se*.

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Principle 1

Quality attribute requirements are those that drive the design of the software architecture

Leads to several questions:

- 1) How are quality attribute requirements specified
- 2) How are quality attribute requirements achieved
- 3) How can understanding of the impact of quality attributes on design be used to improve the design and evaluation processes?

© 2004 by Carnegie Mellon University

page 26

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Characterizing Quality Attributes

Requirements such as "the system shall be modifiable", are meaningless.

What does it mean to say "the system shall be modifiable"?

Is a denial of service attack a security problem, a performance problem, an availability problem, or a usability problem?

© 2004 by Carnegie Mellon University

page 27

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

We have a common form for specification of quality requirements

We use **quality attribute general scenarios**, which are system independent, to guide the specification of quality attribute requirements.

We characterize quality attribute requirements for a specific system by a collection of **concrete quality attribute scenarios**. These are instances of general scenarios.

We use **general scenario generation tables** to construct well-formed general scenarios for each attribute.

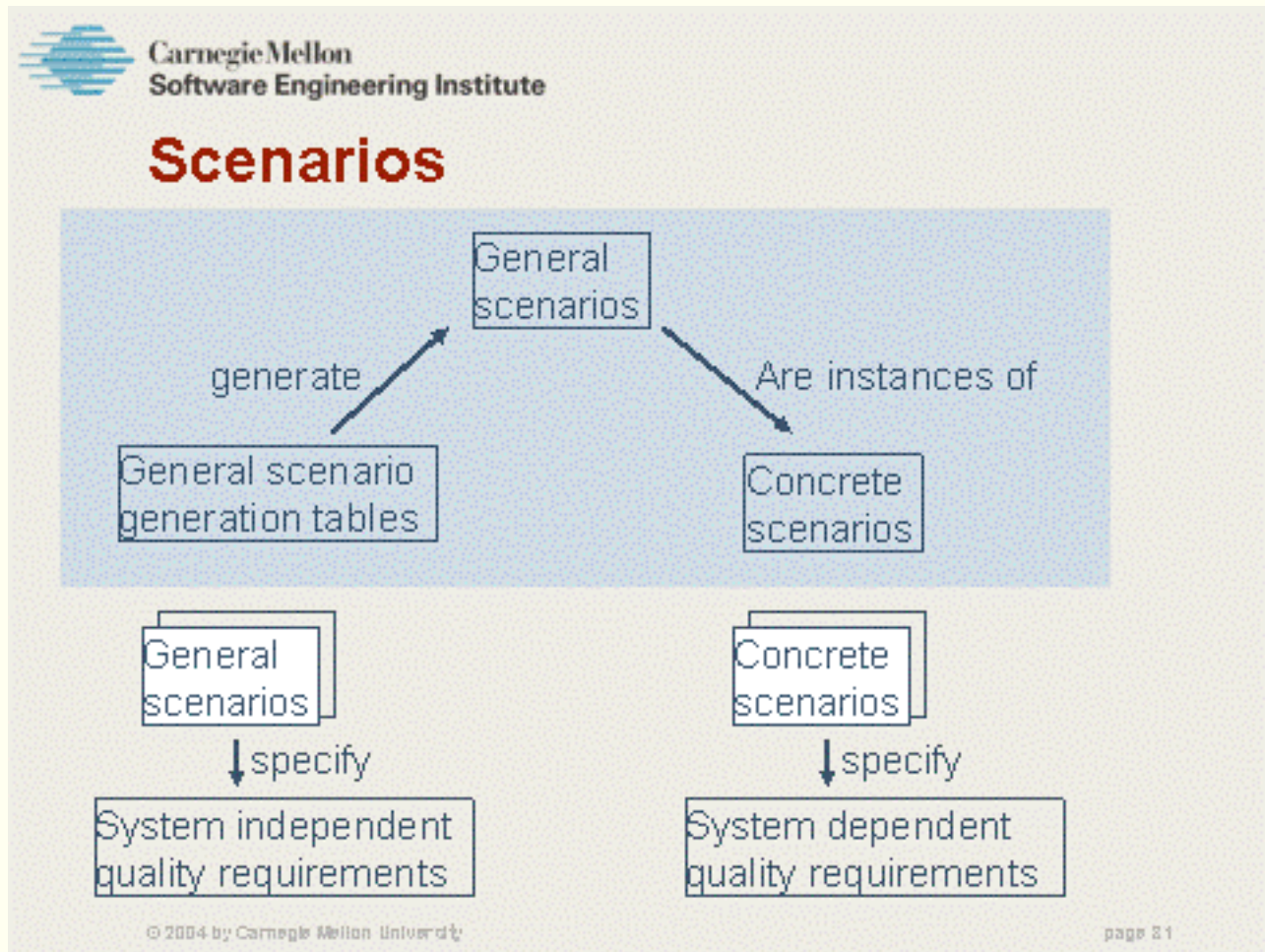
© 2004 by Carnegie Mellon University

page 28

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



[Previous](#) [Next](#)

[Exit Slide Show](#)

- [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



General Scenarios

General scenarios have six parts. The "values" for each part define a vocabulary for articulating quality attribute requirements. The parts are:

- Stimulus
- Source of stimulus
- Environment in which the stimulus arrives
- Artifact influenced by the stimulus
- Response of the system to the stimulus
- Response measures

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Availability Scenario Generation Table

Source of stimulus:

- Internal to the system
- ✓ External to the system

Environment:

- ✓ Normal operation
- Degraded mode

Response:

- ✓ record it
- ✓ notify parties
- ✓ operate in normal or degraded mode

Stimulus:

- ✓ Unanticipated event
- Update to a data store

Artifact:

- ✓ Process
- Persistent storage

Response measures:

- ✓ Availability percentage
- Time range in which the system can be in degraded mode

Example Scenario:

"An unanticipated message is received by a system process during normal operation. The process has to record it, inform the appropriate parties and continue to operate in normal mode without any downtime."

© 2004 by Carnegie Mellon University

page 36

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Constructing Quality Requirements from General Scenarios

Generate a possible general scenario by choosing one or more entries from each list and combining them

Not all:

- general scenarios are relevant to specific system
- generated scenarios make sense

Make each scenario system specific (concrete scenario)

May be multiple concrete scenarios for each general scenario.
e.g., modify function.

Eliminate duplicates

© 2004 by Carnegie Mellon University

page 27

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Which attributes?

We have focused on six quality attributes:

- Availability
- Modifiability
- Performance
- Security
- Testability
- Usability

Others are equally important:

- Buildability
- Interoperability
- ...

© 2004 by Carnegie Mellon University

page 28

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

What about function and quality?

Software for garage door opener

Some scenarios:

- "Halt garage door when an obstacle is detected"
- "respond to user's requests to raise/lower the door within .5 second"
- "replace sensor/actuator within 40 staff hours"

Functional or quality requirements?

© 2004 by Carnegie Mellon University

page 41

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Functional AND Quality

Every requirement has both functional AND quality portions.

E.g. Halt garage door when an obstacle is detected.

Function: detect obstacle, halt garage door

Quality: within time limit (implicit in this example).

Scenario template provides means for eliciting quality requirements associated with functions.

Quality portion leads to design template in which to situate functionality

© 2004 by Carnegie Mellon University

page 48

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Key Ideas of this section

Can express business goals as quality scenarios

Can characterize quality scenarios in structured fashion

For six attributes, we have tables that enable the generation of quality attribute scenarios

Quality attribute scenarios provide quality attribute requirements to particular functions

© 2004 by Carnegie Mellon University

page 46

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Principle 2

Quality attribute requirements can be specified through concrete scenarios with six parts.

Still questions left:

- 1) How are quality attribute requirements achieved
- 2) How can understanding of the impact of quality attributes on design be used to improve the design and evaluation processes?

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)

What does it mean to achieve a quality attribute requirement?

Quality Attribute Requirement

Quality Attribute Model

Software Architecture

A quality attribute requirement defines a region within the set of quality attribute measures.

An architecture can be interpreted in terms of quality attribute model which, in turn, can be evaluated to determine which quality attribute value the software architecture will achieve for a particular stimulus.

Quality Attribute Measure

If evaluated value is inside the region defined by the requirement, the requirement is satisfied.

© 2004 by Carnegie Mellon University page 48

[Previous](#) [Next](#)

[Exit Slide Show](#)

- [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Achieving Quality Attribute Requirements

Design is the process of making decisions about various design options

We can enumerate decisions known to be useful in achieving different quality attributes.

For example: what are some decisions used to improve performance? To improve modifiability? To improve availability?

© 2004 by Carnegie Mellon University

page 61

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



What are these architectural decisions?

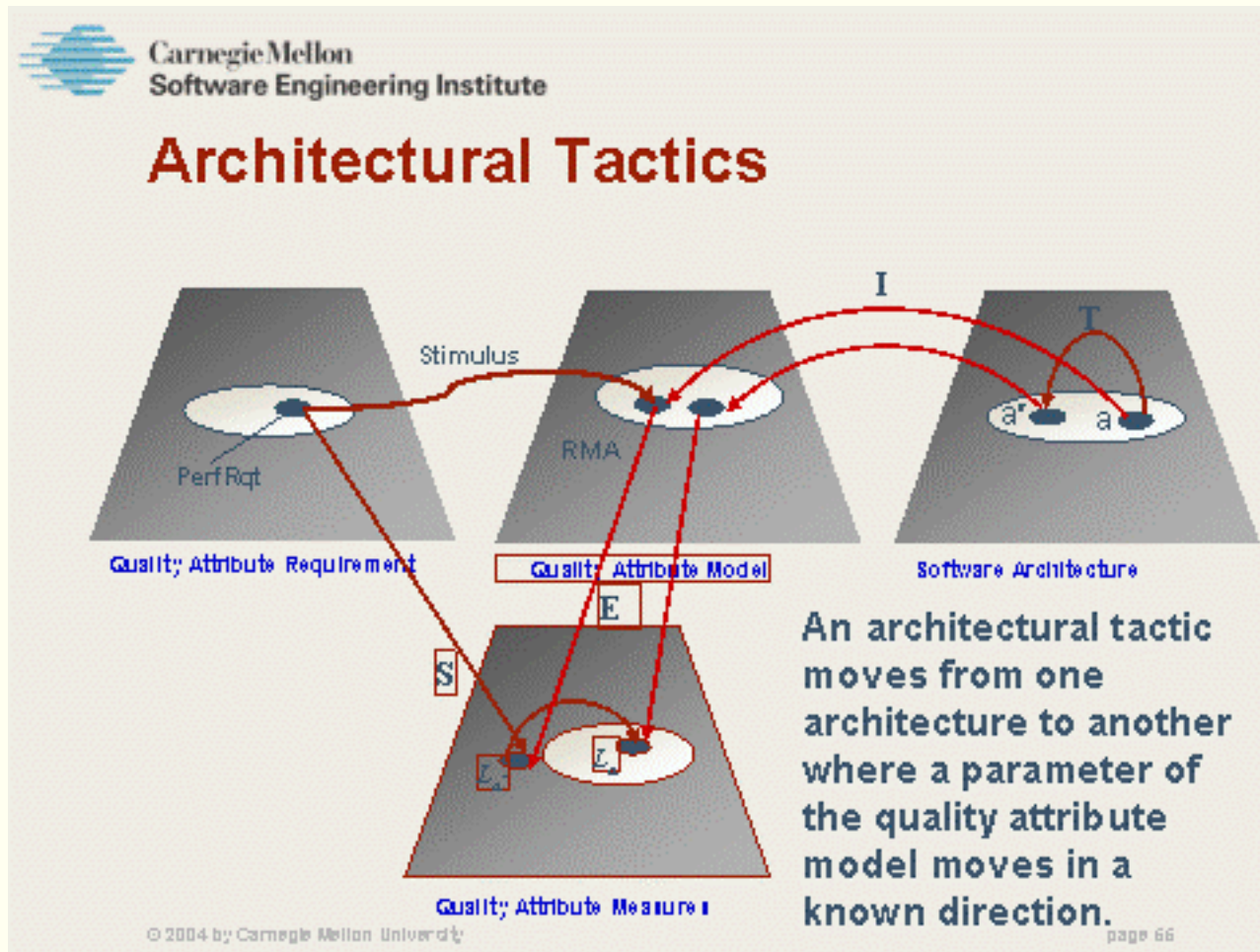
For the six quality attributes –availability, modifiability, performance, security, testability, usability - we have enumerated a collection of “tactics”

Formal definition: *An architectural tactic is a means of satisfying a quality attribute response measure by manipulating some aspect of a quality attribute model through architectural design decisions.*

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



[Previous](#) [Next](#)

[Exit Slide Show](#)

- [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
- [31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Informal Characterization of Tactics

"An architectural pattern is a collection of components and interactions to resolve multiple conflicting forces" –
Buschmann [1996]

An architectural tactic is a transformation that will move in the direction of resolving a single quality attribute force.

© 2004 by Carnegie Mellon University

page 67

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Tactics bridge quality attribute models and architectural design

Tactics identify key quality attribute concepts and bridge quality attribute model and architectural design

- Modifiability model has concepts such as "dependency"
- A tactic for controlling dependency is "use an intermediary"

Quality attribute models (analytic, empirical or qualitative) drive the identification of tactics

- Derived from attribute experts, not necessarily through examination of explicit models
- Derived from well-known analytic models

© 2004 by Carnegie Mellon University

page 68

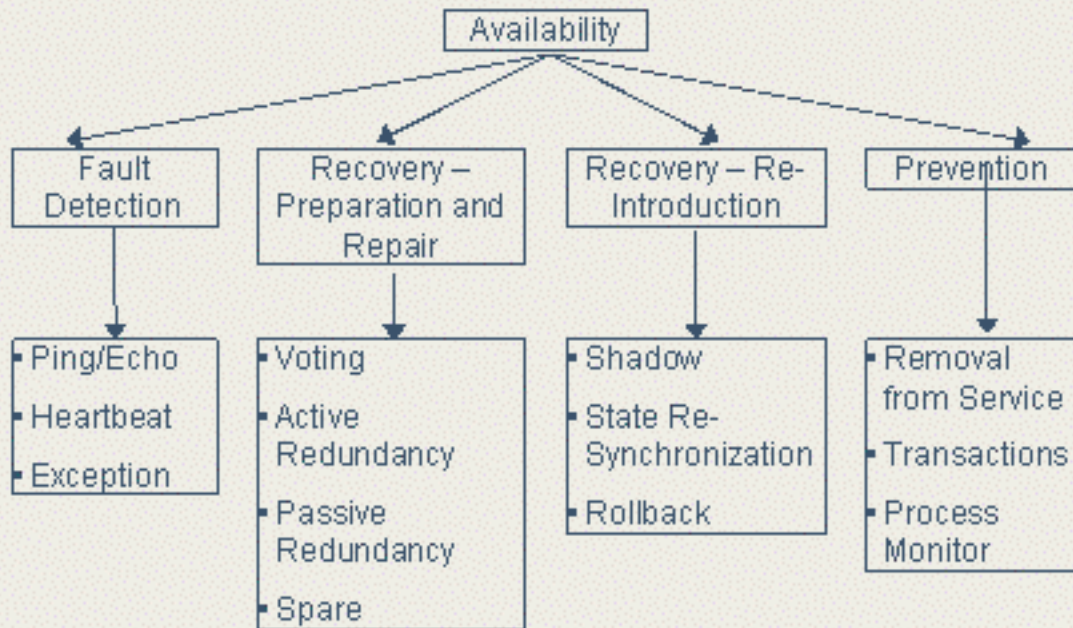
[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Availability Tactics



[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Principle 3

Quality attribute requirements can be achieved through application of architectural tactics

Still questions left:

- 1) How can understanding of the impact of quality attributes on design be used to improve the design and evaluation processes?

© 2004 by Carnegie Mellon University

page 82

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Informal discussion of architectural evaluation

Architectural evaluation – examine existing architecture to determine how well it satisfies its quality attribute requirements.

Ideally limited amount of time is spent on the evaluation

Leads to three problems:

1. Which quality attribute requirements are the focus of the evaluation?
2. Which portion of the architecture is the focus of the evaluation?
3. What do you look for inside the architecture?

© 2004 by Carnegie Mellon University

page 86

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Which quality attribute requirements are the focus of the evaluation?

Want to focus on those requirements that

- are most important for business goals
- have largest impact on architecture

Suggests that input is needed from:

- business people – e.g. marketing
- technical people – e.g. architect

Our method – ATAM™ - has steps that select scenarios to focus on. These steps involve multiple stakeholder representatives including architect and marketing.

© 2004 by Carnegie Mellon University

page 67

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Which portion of the architecture is the focus of the evaluation?

Architecture is large and it is time consuming to examine all of it.

Want to emphasize that portion of the architecture that realizes the scenarios that are the focus of the evaluation.

ATAM uses the architect to identify the portion of the architecture affected by a scenario. The architect walks through how the scenario is achieved.

© 2004 by Carnegie Mellon University

page 88

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) 35 [36](#) [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

What do you look for inside the architecture?

Once focused on a portion of the architecture the evaluator must determine whether any architectural decisions affect business goals.

Evaluator examines the architecture for any business goals, not just ones affected by focusing scenario.

Use of tactics (and lack of use of tactics) are indicators of whether the architecture has a problem achieving a particular business goal.

© 2004 by Carnegie Mellon University

page 7 1

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) 36 [37](#) [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Informal discussion of architectural design

Architectural design is intended to provide high level design of system. Provides structure that enables fulfillment of requirements.

Our design methods exploit knowledge of quality attribute. They use tactics to generate architectural hypotheses that are tested against quality attribute requirements

Methods:

- Attribute Driven Design (ADD). Human centered, does not require special tool support
- Predictable Architecture Design (PAD). Level of detail and breadth of quality attribute knowledge require specialized tool support.

© 2004 by Carnegie Mellon University

page 73

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) 37 [38](#) [39](#)



Carnegie Mellon
Software Engineering Institute

Summary

Quality attribute requirements determine architectural design

Quality attributes requirements can be expressed in a common form

Architectural tactics are an enumeration of techniques that architects use to achieve particular quality attributes

Knowledge of quality attributes can be embodied in design and evaluation methods.

© 2004 by Carnegie Mellon University

page 76

[Previous](#) [Next](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) 38 [39](#)



Carnegie Mellon
Software Engineering Institute

More Information

Lists of general scenarios and tactics, descriptions of ATAM and ADD are available in second edition of

Software Architecture in Practice

<http://sei.cmu.edu/architecture>

Software architecture curriculum

Len Bass: ljb@sei.cmu.edu



[Previous](#)

[Exit Slide Show](#)

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)
[31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) 39