

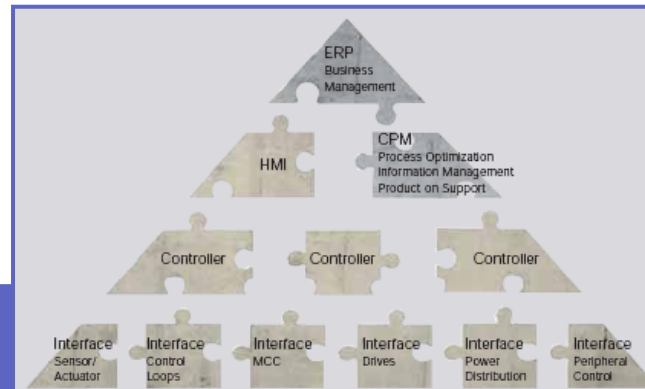
Defining Composite Critical Scenarios for the Development of Large Scale System Architecture Using an SEI's ADD-based Framework

Aldo Dagnino

ABB Inc.

**US Corporate
Research Center**

Raleigh, NC



SATURN 2008

**Fourth SEI Software Architecture Technology
User Network Workshop**

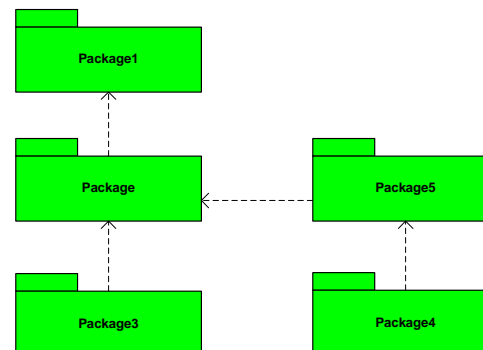
April 28-May 1 of 2008

Pittsburgh, Pennsylvania

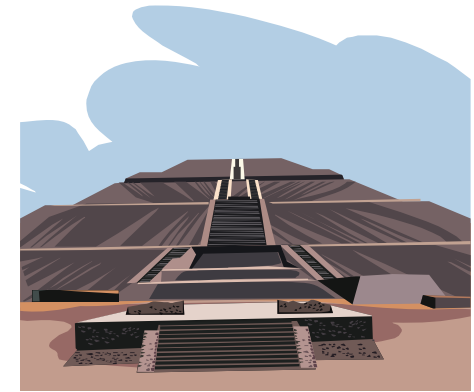
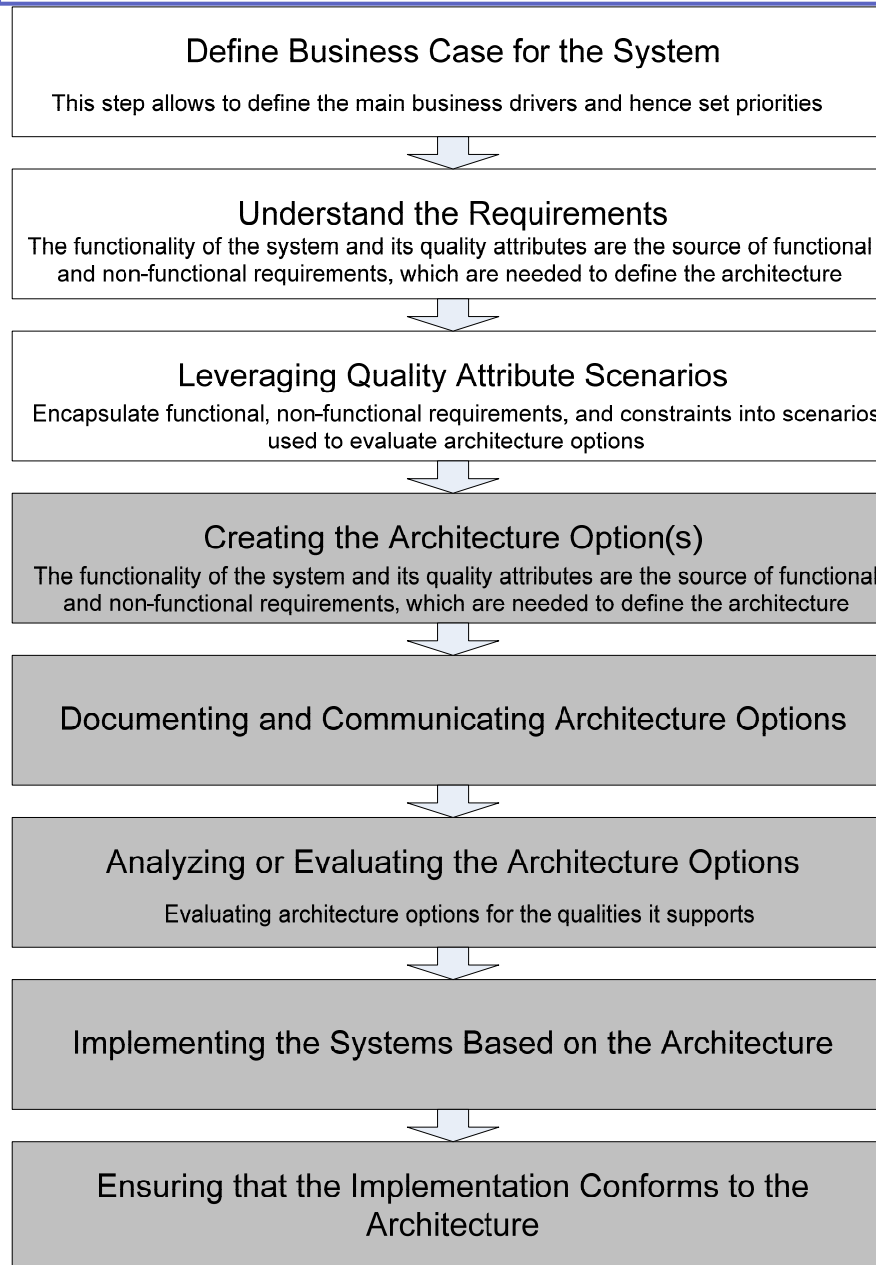


Attribute Driven Design (ADD)

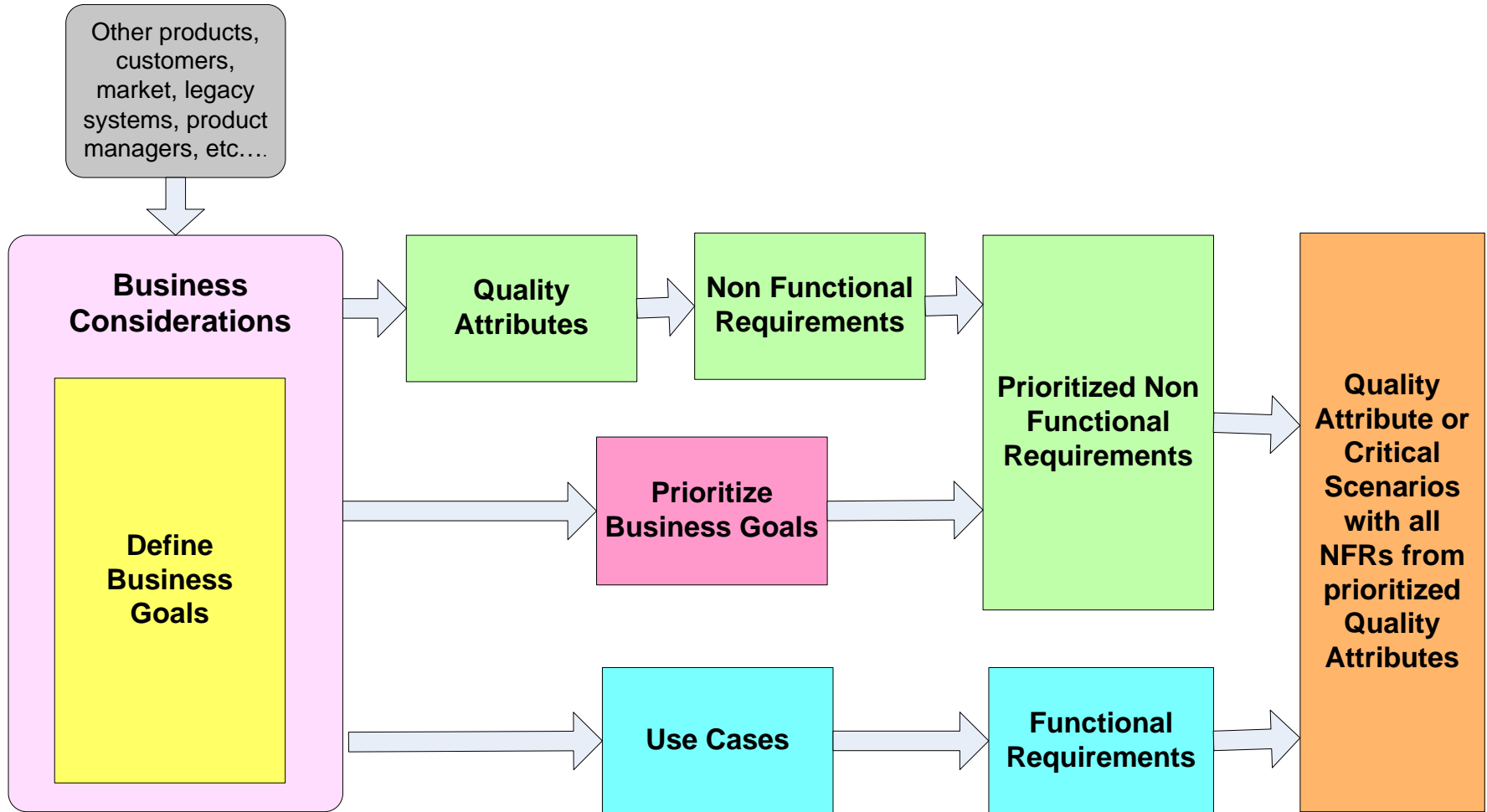
- ADD is a methodology used to define a system architecture that bases the decomposition process on the quality attributes the system (software) has to fulfill.
- The architectural design using the ADD methodology can begin when the architectural drivers are known with some level of confidence.
- In ADD Tactics and Architectural patterns are selected to satisfy a set of quality attributes within a critical scenario that provides context for those quality attributes



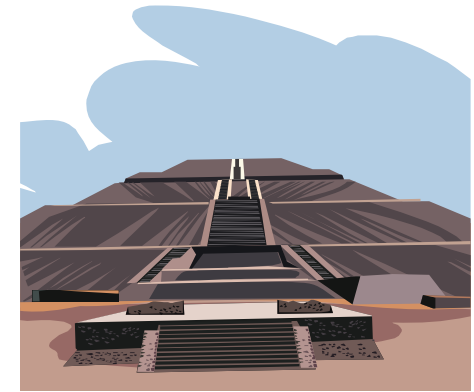
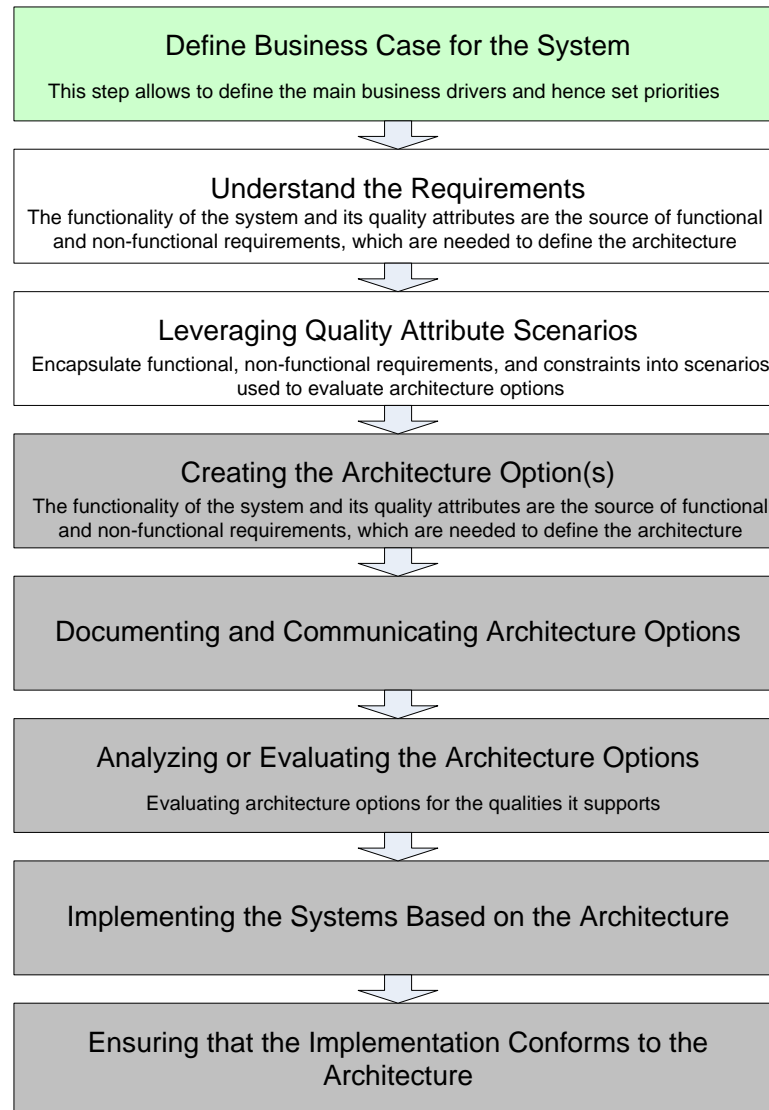
Steps for Creating a Software Architecture



Prioritizing Business Goals



Define Business Case



Business Goals

■ Prioritized Business Goals

- Business goals associated with the project are elicited from selected project stakeholders
- Business goals are prioritized for stakeholders to guide architectural tradeoffs



■ Example of prioritized business goals:

- Lower commissioning costs by 30%
- Ensure system is available 99.99%
- Maintain current system performance

Mapping Business Goals and Quality Attributes

Business Goal

Quality Attributes

Lower commissioning costs by xx%

Commissionability

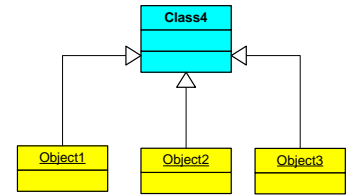
Ensure system is available 99.9%

Availability

Maintain current system performance

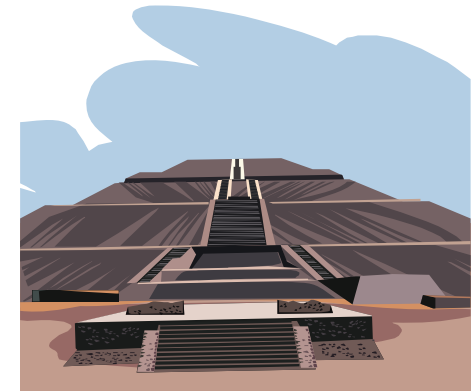
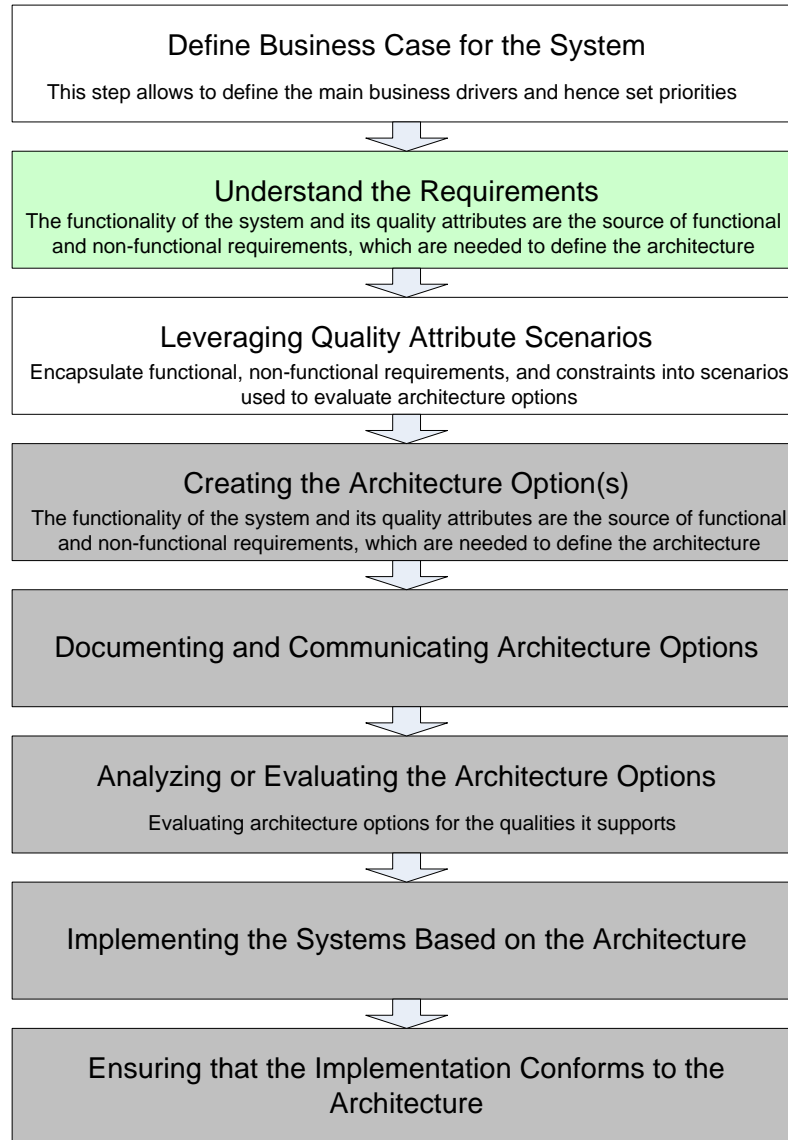
Performance

Business Goals and Quality Attributes



Business Goals	Software Qualities
Lower commissioning (customization, installation, and configuration) and recurring system setup costs by xx% in new System	Commissionability
Ensure data integrity in system	Data Integrity
Ensure System is available 99.99%	Availability
Maintain current system performance	Performance

Understand Requirements



Develop Non-Functional Requirements

Quality Attribute
System Quality

Customer-related
Non Functional Requirements
Associated/derived from
Quality Attribute

Commissionability

```
graph LR; A[Commissionability] --> B[NFR0250 - No source code change for a specific customer deployment]; A --> C[NFR0260 - Complete software build (compilation of source code) within x hours]; A --> D[NFR0270 - Complete system installation (including OS) within y hours (human interaction)];
```

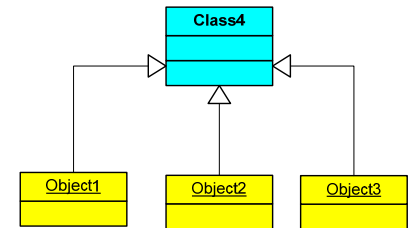
NFR0250 - No source code change for a specific customer deployment

NFR0260 - Complete software build (compilation of source code) within x hours

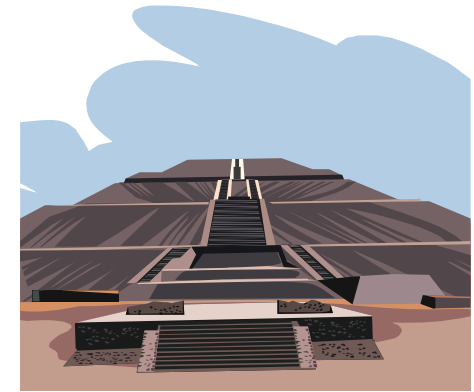
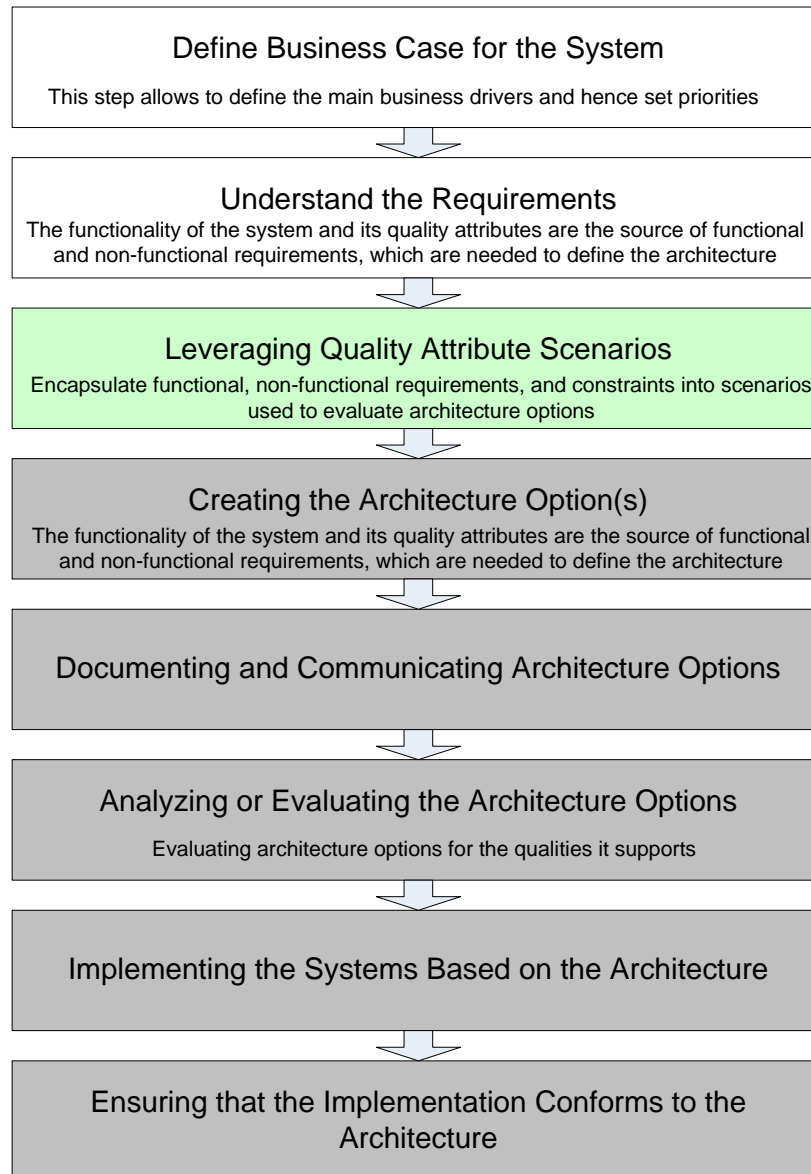
NFR0270 - Complete system installation (including OS) within y hours (human interaction)

Quality Attributes and Non-Functional Requirements

Business Goals	Driving Non-Functional Requirements	Software Qualities
Lower commissioning (customization, installation, and configuration) and recurring system setup costs by xx% for new system	<p>NFR0250 - No source code change for a specific customer deployment</p> <p>NFR0260 - Complete software build (compilation of source code) within x hours</p> <p>NFR0270 - Complete system installation (including OS) within y hours (human interaction)</p>	Commissionability
Ensure 100% data integrity in system	<p>NFR0110 - Ensure data integrity across integrated systems</p> <p>NFR0100 - Provide controlled access to its data stores</p> <p>NFR0030 - Prevent data corruption and data loss</p>	Data Integrity
Ensure system is available 99.9%	<p>NFR0750 - Be available 24*7 (RFPs: 99.9%)</p> <p>NFR0720 - Failover for XXX and YYY functions shall be completed within xx seconds</p> <p>NFR0630 - The architecture must support hot failover</p> <p>NFR0740 - Loose coupling of components</p> <p>NFR0700 - Complete cold startup (from power-off condition) shall be completed with all functions scheduled for execution within xx time</p>	Availability
Maintain current system performance	<p>NFR0820 - Events processing speed</p> <p>NFR0870 - Display update upon change max exec time is xx sec</p> <p>NFR0030 - Prevent data corruption and data loss</p> <p>NFR0890 - Process events</p> <p>NFR0910 - Execution time of applications</p> <p>NFR0900 - Time to run sequence of applications from Topology Processing to Contingency Analysis</p> <p>NFR0860 - Supervisory control max execution time is 1 sec</p> <p>NFR0920 - CPU and memory consumption of applications</p>	Performance

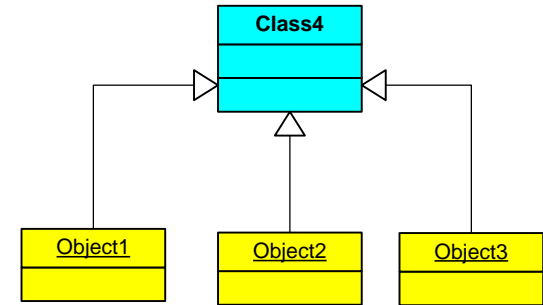


Leveraging Quality Attribute Scenarios

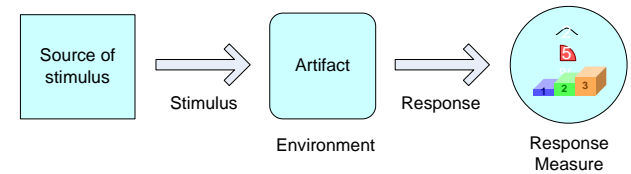


Architectural Drivers

- Architectural drivers (quality attribute scenarios) include the combination of functional and quality requirements that shape the architecture:
 - Define unique functions (as architectural Functional Requirements) of modules in the system
 - Select associated Non-functional Requirements
 - Quality Attributes or Critical System Scenarios provide the functional context under which Non Functional Requirements are defined
 - Architectural patterns that satisfy the critical scenarios are then selected

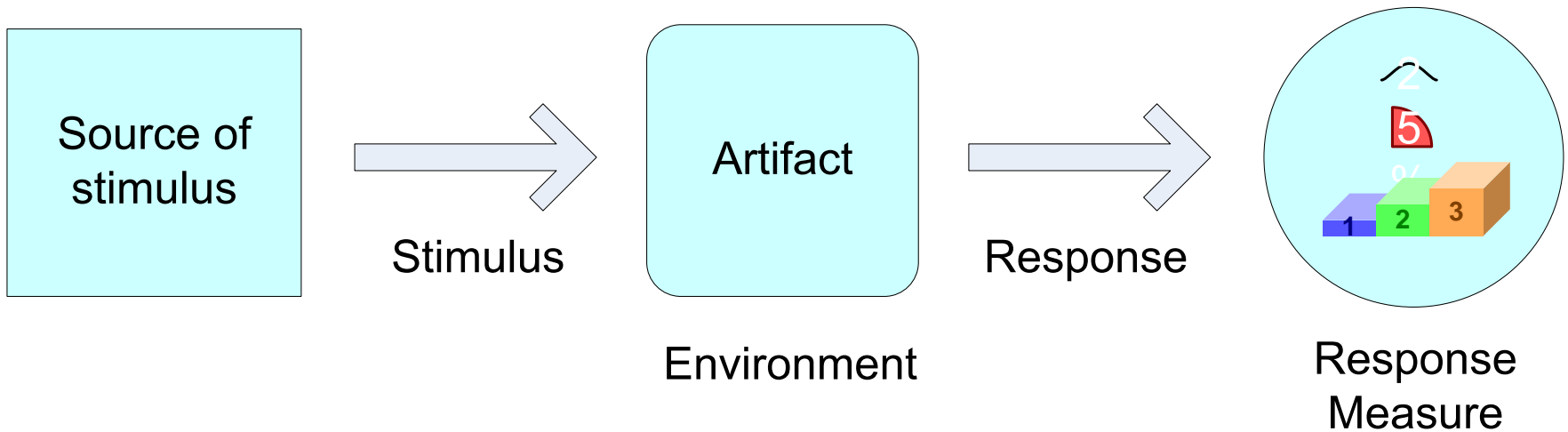


Quality Attribute Scenarios

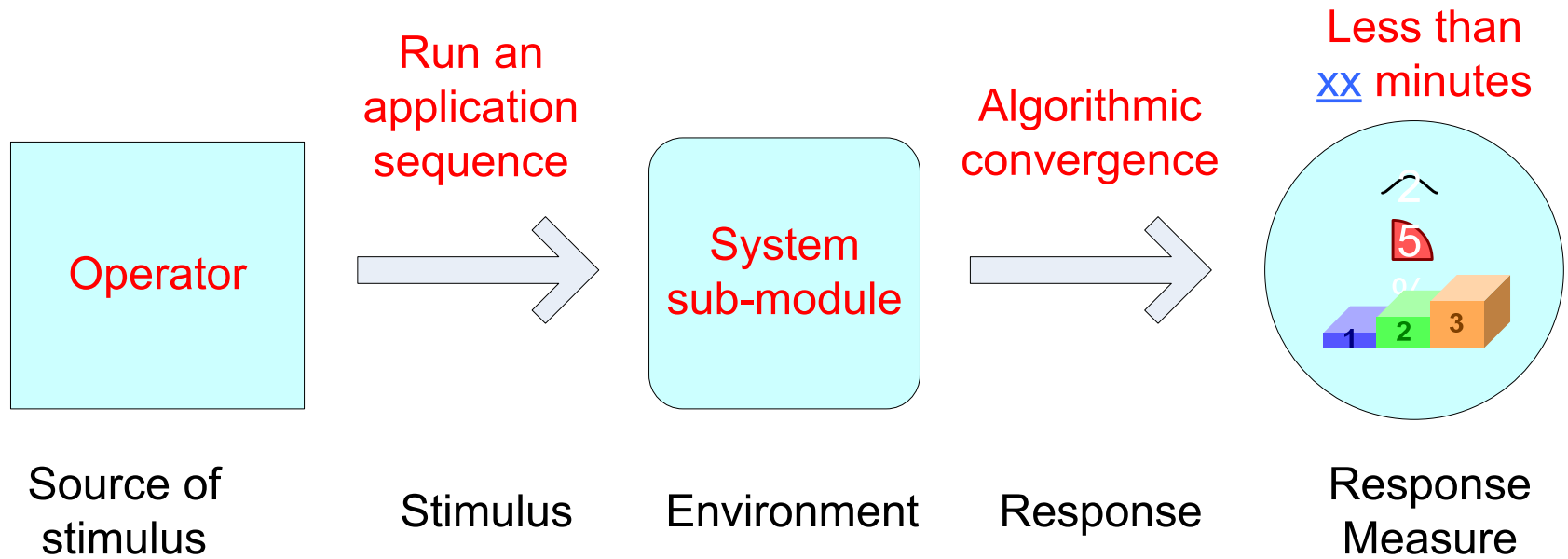


- **Encapsulate a set of architectural functional and non-functional requirements that uniquely define the system being architected**
- **Are described by a set of detailed architectural product requirements**
- **Can incorporate of one or more Use Cases**

Quality Attribute Scenario Elements



Quality Attribute Scenario: Run a Sequence of Applications



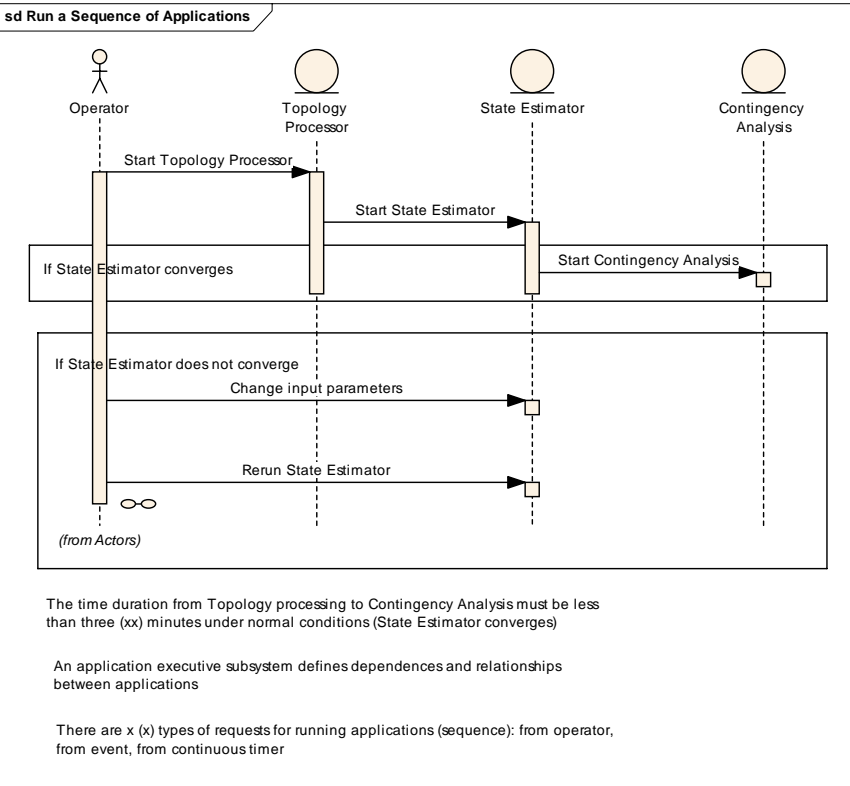
Develop Critical System Scenarios

Critical Scenario

The operator runs a sequence of complex applications

Customer (Architectural) Requirements

Includes Functional and Non-functional requirements (and any constraints)



The system shall allow the operator to run the state estimator application

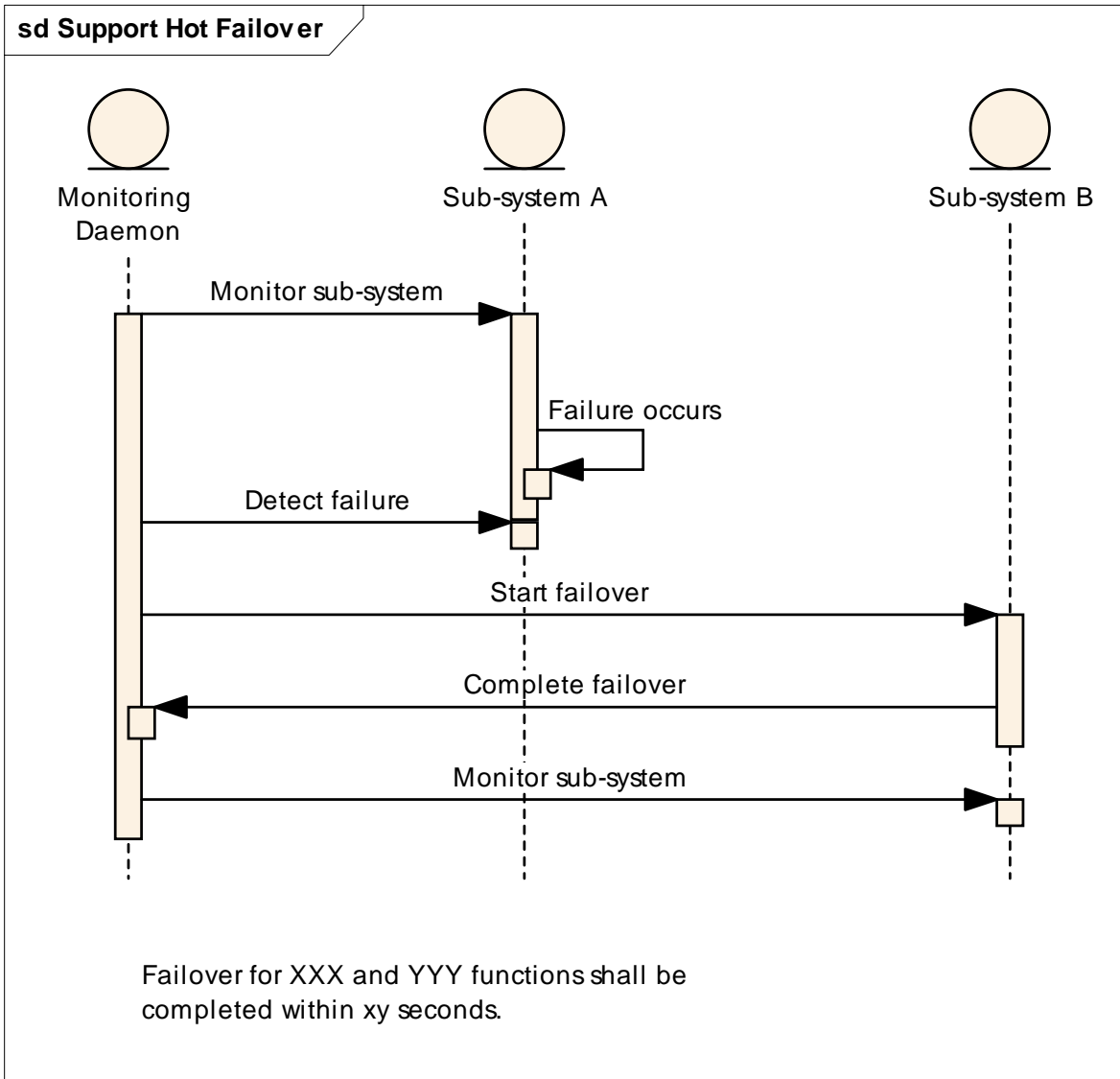
The system shall allow the operator to run sensitivity analyses

The system shall allow the operator to run the PS model

The system shall allow the operator to run a sequence of applications in an "industry acceptable" time

etc . . .

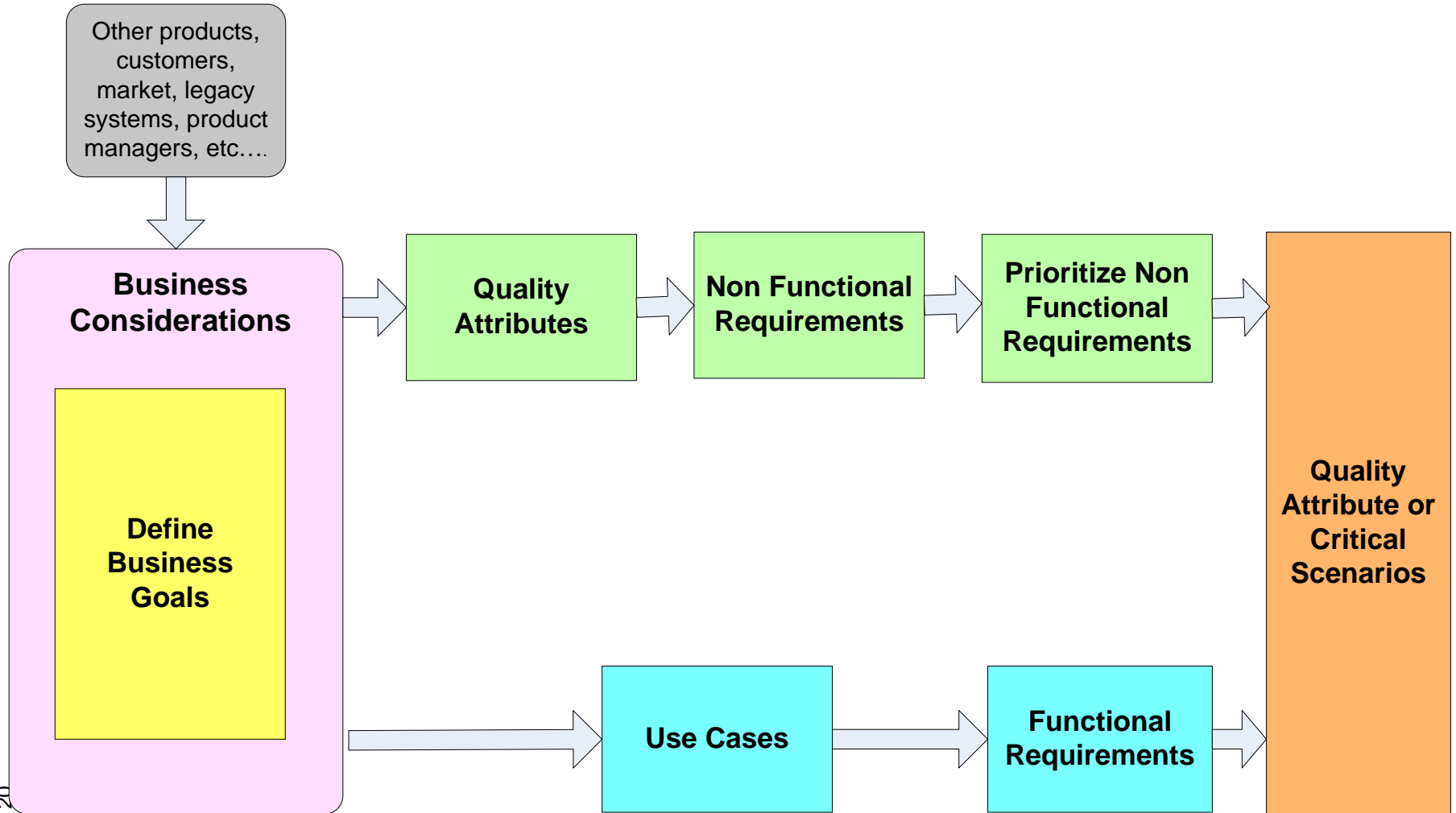
Another Critical System Scenario



Critical System Scenarios

- **Add new element and sub-elements to the system**
- **Integrate with external systems**
- **Process and display events**
- **Run advanced numerical applications**
- **Support hot-failover**
- **Run a sequence of complex applications**

Prioritizing Non-Functional Requirements



Prioritized Non-functional Requirements

ID	Requirement	Calculated Avg.	Impact on Architecture	Prototype Importance	10*CalcAvg +ImpactArch
4.2.1.1	The new system will have one code base to ensure the functionalities of XXX system by using common technology and programming language, as well as shared or reusable modules.	5.0	5	5	55
4.2.4.1	The new system will have consistent mechanisms for integration with other components of overall system.	5.0	5	3	55
4.6.1.1	The failover of a server component using a backup database completes in no more than xx seconds.	5.0	5	1	55
4.3.3.2	The system allows the user to perform only the authorized actions.	5.0	5	1	55
4.2.5.4	The implementation plan shall facilitate the transition of the XXX team to the new development environment.	5.0	5	0.1	55
4.2.1.2	The impact of changes will be minimized in the new system thanks to a modular design	5.0	3	3	53
4.1.3.2	Shortcuts to achieve multi-row operations are available (aggregated data). Mass-update.	5.0	3	3	53
4.2.4.2	The new system will allow for interactive exchange of data between new YYY module and ZZZ module (messaging layer).	4.7	5	5	52
4.4.1.3	Dialog response time is less than x seconds in average and never more than y seconds.	4.7	5	5	52



Develop Quality Scenarios Using
Prioritized NFRs





- It is easier for Senior Management to prioritize Business Drivers than non functional requirements
- Creation of Composite Scenarios provides a richer approach to prototype architectural options
- Prioritization of business goals automatically prioritizes qualities and non functional requirements
- Potential for missing important non functional requirements that fall outside the prioritization scheme
- Analysis may become more complex using Composite Scenarios



Power and productivity
for a better world™