Architecture Development Leveraging the Attribute Driven Design and the CMMI Methodologies

Dr Aldo Dagnino
ABB Inc. US Corporate Research Center

CMMI Technology Conference and User Group
November 12-15, 2007
Hyatt Regency Tech Center, Denver CO
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

- Creating the business case for the system
- Understanding and documenting the requirements
- Leveraging Quality Attribute Scenarios
- Creating or selecting the architecture
- Documenting and communicating the architecture
- Analyzing or evaluating the architecture
- Implementing the system based on the architecture
- Ensuring that the implementation conforms to architecture
Integration of ADD and CMMI

Business Considerations
- Define Business Goals
- Prioritize Business Goals

Quality Attributes

Non Functional Requirements

Use Cases

Functional Requirements

Software Functionality

RD and REQM CMMI Process Areas

Architecture Tactics

Quality Attribute Scenarios

Other products, customers, market, legacy systems, product managers, etc.
Steps for Creating a Software Architecture

- Creating the business case for the system
- Understanding and documenting the requirements
- Leveraging Quality Attribute Scenarios
- Creating or selecting the architecture
- Documenting and communicating the architecture
- Analyzing or evaluating the architecture
- Implementing the system based on the architecture
- Ensuring that the implementation conforms to architecture
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 xx%
- Ensure system is available 99.9%
- Maintain current system performance
- etc
Mapping Business Goals and Quality Attributes

Business Goal

- Lower commissioning costs by xx%
- Ensure system is available 99.9%
- Maintain current system performance

Quality Attributes

- Commissionability
- Availability
- Performance
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 attribute scenarios provide the functional context under which Non Functional Requirements are defined
- Architectural patterns that satisfy the critical scenarios are then selected
Steps for Creating a Software Architecture

- Creating the business case for the system
- Understanding and documenting the requirements
- Leveraging Quality Attribute Scenarios
- Creating or selecting the architecture
- Documenting and communicating the architecture
- Analyzing or evaluating the architecture
- Implementing the system based on the architecture
- Ensuring that the implementation conforms to architecture
Develop Customer (Architectural) Requirements

SP 1.1 Elicit needs

SP 1.2 Develop the customer (architectural) requirements

Use Case
The operator runs a sequence of complex applications

Customer (Architectural) Requirements
Includes Functional and Non-functional requirements

- 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 . . .

etc . . .
SP 1.1 Elicit needs

SP 1.2 Develop the customer (architectural) requirements

**Quality Attribute**
System Quality

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

- The source code for the system shall not be modified for any customer implementation
- The software build shall be completed in an “acceptable” time period
- The complete system installation shall be completed in an “acceptable” time period

Commissionability
SP 2.1 Establish product and product component requirements
SP 2.2 Allocate product component requirements
SP 2.3 Identify interface requirements

**Customer Requirements**
Includes Functional and Non-functional requirements

**Product Architectural Requirements**
Testable and measurable set of requirements

The system shall allow the operator to run the state estimator application.
The system shall allow the operator to run the state estimator application in xx seconds.

The system shall allow the operator to run sensitivity analyses.
The system shall allow the operator to run sensitivity analyses in yy seconds per run.

The system shall allow the operator to run the PS model.
The system shall allow the operator to run the PS model in xy seconds.

The system shall allow the operator to run a sequence of applications in an “industry acceptable” time.
The system shall allow the operator to run a sequence of applications in yz seconds.
Steps for Creating a Software Architecture

- Creating the business case for the system
- Understanding and documenting the requirements
- Leveraging Quality Attribute Scenarios
- Creating or selecting the architecture
- Documenting and communicating the architecture
- Analyzing or evaluating the architecture
- Implementing the system based on the architecture
- Ensuring that the implementation conforms to architecture
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

Source of stimulus -> Stimulus -> Artifact -> Environment -> Response

Response Measure
Analyze and Validate Requirements

- **SP 3.1** Establish operational concepts and scenarios
- **SP 3.2** Establish a definition of required functionality
- **SP 3.3** Analyze requirements
- **SP 3.4** Analyze requirements to achieve balance
- **SP 3.5** Validate requirements

**Quality Attribute Scenario**
Sequence Diagram

- **Detailed Architectural Non Functional Requirements**
  Placed in context of Critical Scenario

The time duration of sequence calculations shall be less than xx seconds under normal loading conditions.

The performance of running the numerical application sequence shall be such that it will not exceed specified bounds of memory and CPU load capabilities.
SP 1.1 Obtain an understanding of requirements
SP 1.2 Obtain commitment to requirements
SP 1.3 Manage requirements changes
SP 1.4 Maintain bi-directional traceability of requirements
SP 1.5 Identify inconsistencies between project work and requirements

Understanding and commitment to requirements among stakeholders carried out through meetings

Functional and Non Functional requirements
Stored, managed, and maintained in Enterprise Architect and Requisite Pro tools
Attribute Scenario: Run a Sequence of Applications

Source of stimulus  Operator  Run an application sequence  System sub-module  Algorithmic convergence  Response Measure

Stimulus  Environment  Response

Less than $xx$ minutes
Lessons Learned

- The practices of the RD process area greatly contribute to defining the functional and non-functional architectural requirements that form the basis for ADD.
- Organization business objectives are essential to establish priorities that drive the development of the architecture.
- Quality attribute scenarios provide context to non-functional requirements.
- To implement quality attribute scenarios, specific tactics identified in ADD provide architectural patterns.
ABB

Power and productivity for a better world™