Application Servers
Session 2 – Main Theme
Architectural Mapping
Dr. Jean-Claude Franchitti
New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Agenda

1. Introduction
2. Architecture Blueprinting
3. Sample Architecture Blueprints
4. Architectural Mapping Process Illustrated
5. Conclusion
Architecture Models

- An Architecture provides the organizing logic for mapping business onto IT capabilities
- Creating models to describe an Architecture is a complex exercise as various levels of abstractions may need to be considered to effectively cover all requirements in increasing levels of detail
- Architecture Models are typically based on an integration of existing reference architecture styles
  - e.g., OMA and SOA, SOA and BPM, etc.
A Reference Architecture consists of foundational principles, an organizing framework, a comprehensive and consistent method, and a set of governing processes and structures.

- Principles provide the foundation upon which the Reference Architecture is based. It includes a set of architectural terms as well as numerous principles, policies, and guidelines for governing the architecture.
- Framework is the organizing basis for the Reference Architecture and defines the architectural domains and disciplines that enable separation of concerns and IT to business alignment.
- Method is the comprehensive set of defined repeatable processes that are followed for a consistent and controlled realization of the Reference Architecture.
- Governance is the set processes and organizational structures that ensure conformity to the Reference Architecture.

Sample Application Server Reference Architecture

JBoss Developer Studio
- Eclipse IDE
- Integrated building (plugins)
- Embedded runtime platform
- Fully integrated development environment

JBoss Enterprise Portal Platform
- Content aggregation, presentation and personalization

JBoss Enterprise Application Platform
- JBoss Enterprise Web Platform
- JBoss Enterprise Web Server
- Application and service container, data persistence, messaging and transactions

JBoss Enterprise SOA Platform
- JBoss Enterprise BRMS
- Service integration/orchestration, business process automation, rules definition and event management

JBoss Enterprise Data Services Platform
- Data integration, data services federation, data abstraction and management

JBoss Operations Network
- Administration, management, and monitoring
Reference Architecture Models

- A “reference” architecture model is an accepted representation of the architecture that drives the mapping of business capabilities onto IT capabilities.
- A reference architecture model may not represent any specific organization needs.
- A reference architecture model is rarely developed using a top-down or bottom-up approach, it is typically put together by integrating requirements from various architectural domains according to accepted heuristics (e.g., reuse via unification or best practices / standardization) and using accepted frameworks.

Enterprise Architecture Modeling Heuristics

### Coordination
- Shared customers / products / product data / suppliers
- Operationally autonomous and unique business units
- Transactions impact other business units

### Unification
- Customers and suppliers may be local or global
- Business units with similar or overlapping operations
- Globally integrated business processes supported by Enterprise systems

### Diversification
- Few, if any, shared customers or suppliers
- Operationally autonomous and unique business units
- Independent transactions

### Replication
- Few, if any, shared customers
- Operationally similar business units
- Independent transactions aggregated at a higher level
- Autonomous business units with limited discretion over processes

Business process integration: High
- Standardization

Optional: Required

High

----------------------

Business process standardization

----------------------

Optional

High
Architects working at the Enterprise, Portfolio, and System levels use different models to represent their own views of a given architecture.

An Architecture Asset Catalog enables the representation of stakeholder views in matrix form to help catalog such models.

What is the Level of Abstraction?

The level of abstraction refers to how far a blueprint is removed from “practical” considerations such as application servers, programming languages, DBMS technology, etc. Although the levels of abstraction vary by architecture domain, four different types levels are recognized:

- **Presentation** - a stylized model intended to greatly simplify an architecture so that key messages can be effectively communicated, typically to business leadership.
  - Presentation level diagrams are generally created by summarizing lower level architecture diagrams.

- **Conceptual** - a highly generalized yet more formal depiction of an architecture that suppresses much of the actual detail — either because the details are not important to the model and/or they had not been decided at the time the diagram was created.

- **Logical** - a detailed representation of an architecture that is generally independent of the underlying technology that is used (or will be) used to implement an architecture.

- **Physical** - A representation that is typically dependent on the underlying technology that will be (or is) used to implement an architecture.
In this context, relevant views are those that match the phases of an solution development life cycle.

**Conceptual business architecture framework**

- Value Chain
- Sales and Marketing
- Product Development
- Underwriting
- Finance and Accounting
- Servicing
- Claim Processing
- Business Capabilities
- Business Users
- Business Events
- Business Channels
- BPM Processes
- Process metrics
- BRM Rules
- Business Services
- Business Entities

- Surety
- Management Liability
- Enterprise
- Claim
- Business Unit
Conceptual Business Architecture Framework Illustration

### Agenda

1. **Introduction**
2. **Architecture Blueprinting**
3. **Sample Architecture Blueprints**
4. **Architectural Mapping Process Illustrated**
5. **Conclusion**
What is Blueprinting?

- Blueprinting is fundamentally concerned with the high-level representation of intangible assets (e.g., applications, databases, interfaces, networks, servers, etc.) so that:
  - The interrelationship between the various assets can be understood
  - The assets may be changed more reliably
  - Architectural level design decisions become observable

What is a Blueprint?

- A blueprint is an architectural drawing
  - Created using a consistent representation to represent a high-level model of the as-it, to-be, or in-transition IT environment
- Unlike UML models, which are software engineering level diagrams, blueprints are at an architectural-level of detail and provide the context needed to visualize the “big picture”
  - As such, blueprints are analogous to the “city-planning” level in the building construction industry
  - They enable architects to communicate the overall design of the city as opposed to the design of the individual buildings that make up the city
What Does a Blueprint Look Like and How is it Used?

- The appearance of a blueprint varies considerably depending upon a number of factors including:
  - The architectural domain being modeled (e.g., application architecture versus technical architecture)
  - The scope of the blueprint (e.g., Enterprise, Portfolio, Project)
  - The level of abstraction (e.g., Presentation, Conceptual, Logical, Physical)
  - The communication objectives of the model
- Blueprints are also used to document and define three different states of technology evolution
  - A current state called the As-Is or POD
  - A future state called the To-Be or POA (typically 12 - 24 months out)
  - One or more transition states, each one called a Transition or planned landing point between the as-is and to-be state
    - Once implemented, a Transition represents a new As-Is state

Why is there a Need for a Common Way to Document Architectures?

- In the absence of standardized blueprinting techniques, architectural models would be highly individualized and would range from artifacts that may be fairly structured to models that would be very general and stylistic
- As a result, the readers interpreting the models would be required to ask (and assume an answer to) a number of critical questions including:
  - What concepts is the model attempting to explain?
  - Are the concepts highly abstract or is the model depicting a precise design?
  - What do the symbols on the diagram represent?
  - What architecture domain is being modeled?
  - Does the design apply to the Enterprise as a whole, a LOB, a portfolio, or a project?
  - Does the model represent the As-Is, To-Be, or Transition architecture?
  - If the model represents a Transition architecture, what changes to the IT environment are being planned?
  - etc.
Blueprinting and UML at a Glance

- Blueprinting and UML are intended to be used together on the same project.
- Blueprint artifacts are used to document the end-to-end high-level designs for projects.
  - Blueprints are analogous to the "city-planning" level in the building construction industry.
  - They enable architects to communicate the overall design of a city (project) as opposed to the design of the individual buildings (applications) that make up the city.
- UML artifacts are used for software engineering tasks (e.g., architecting the buildings).

<table>
<thead>
<tr>
<th>Focus</th>
<th>UML</th>
<th>Blueprinting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use</td>
<td>Analyze and design software systems and modules, typically using an OO approach</td>
<td>Describe or prescribe an end-to-end design without delving into details</td>
</tr>
<tr>
<td>Level of detail</td>
<td>High to low-level</td>
<td>High-level</td>
</tr>
<tr>
<td>Central Element of Granularity</td>
<td>A system and its subsystems</td>
<td>System of systems</td>
</tr>
<tr>
<td>Learning Curve</td>
<td>Significant</td>
<td>Minimal</td>
</tr>
</tbody>
</table>

Lack of Blueprinting Standards

- According to Research Analysts and reports...
  - Modeling at the Enterprise and Portfolio levels tends to be fairly generalized.
    - The goal at these levels is to communicate the "big picture" (as opposed to "application-level" designs).
  - UML is an OO modeling system with schematics and notations for application development (e.g., "building-level" designs).
    - It is not well suited for modeling portfolio and Enterprise level architectures (e.g., the "city-level" or "big picture" designs).
  - There may never be industry standards at the Enterprise and Portfolio levels.
The Legend Box is a text box that must appear on all blueprints. It is used to denote important information that is needed by the reader to correctly interpret a blueprint. The following information is included in the Legend Box:

- **Architecture Domain** - Used to specify what aspect of the environment is the subject of architecture artifact - One of the following domains must be specified:
  - *Business Architecture* — specify this when the model depicts the company’s business capabilities, business processes, organizational structure, major locations, or relationships with partners and customers
  - *Application Architecture* — specify this when the model depicts the application assets that support business capabilities and processes
  - *Data Architecture* — specify this when the model depicts the company’s business rules, business data and/or information types, along with their interrelationships
  - *Technical Architecture* — specify this when the model depicts hardware and facilities, system software, data storage resources, networks, and other underlying technologies
  - Technical architecture provide the platform that supports the activities and interfaces of the other domains

- **Scope** - Defines the breadth (or scope of authority) for a blueprint. Several different scopes are recognized:
  - *Enterprise* - A model that generally depicts a company’s environment as a whole
  - *Portfolio* - A model that depicts the architecture of a portfolio (e.g., Field Management)
  - *Program/Project* – A model that depicts the architecture of a program or project
  - *Asset* - A diagram that depicts the architecture of an asset

- **Abstraction** - Refers to how far the model is removed from “practical” considerations such as application servers, programming languages, etc
  - Four different levels are recognized: Presentation, Conceptual, Logical and Physical

- **State** – Used to answer the question: *Does this model represent the current state or some proposed future state?* Three different states are typically recognized:
  - *As-is* - the current state.
  - *To-be* - the desired future state that is to be achieved in a specified time period (typically 12 – 24 months). In reality, the to-be state is a moving target that generally represents an aspiration, as opposed to a fixed target that will be achieved
  - *Transition* - a planned landing point between the current state and the to-be state
  - A Transition diagram shows progress towards the future state
  - Once implemented, a transition architecture represents the new As-Is and the previous current As-Is becomes the As-Was
Importance of the Scope of a Blueprint

- Specifying the scope of a diagram is critical because there is a direct correlation between the scope and the amount of detail that can be depicted on a blueprint. The reason is that the amount of generalization (e.g., simplification, feature selection, grouping, etc.) must increase with the scope of the blueprint. The following examples illustrate this key point:

### Blueprints
- **Enterprise Scope**:
  - As-is Application Architecture for App D
  - Daily Extract

- **System/Project Scope**:
  - As-is Application Architecture for Pgm 1
  - App A
  - App B
  - XYZ Comp

- **Portfolio Scope**:
  - As-is Application Architecture for App Port "C"

### Map Analogy
- **Scope**:
  - United States
  - State of Minnesota
  - City of Minneapolis
  - Downtown Minneapolis

Architecture Blueprinting

Sample Architecture Blueprints

Architectural Mapping Process Illustrated

Conclusion

Agenda
Sample Enterprise Architecture Blueprint for Unification Operating Model

**Delta Air Lines’ Enterprise Architecture**

Customer Experience

Source: Adapted from Delta Air Lines documents - used with permission.

Sample Enterprise Architecture Blueprint for Diversification Oper. Model

**Enterprise Architecture for Carlson’s Diversification Operating Model**

Customer Requirements

Source: Carlson Company
Sample High-Level Business Architecture Blueprint

Conceptual Technology Architecture Blueprint
Mapping Dimensions to Consider

- Levels of abstraction
- Breadth (i.e., architectural domain)
- Depth (i.e., services/facilities needed)
- Specialization (i.e., styles and related pattern)
- Integration of various patterns results in integration variants/hybrids
- Mapping relies on the selection of standards and products that implement that standard
  » e.g., JEE – IBM WebSphere Application Server

Sample Reference Logical Application Architecture Blueprint
(OMA / SOA Hybrid)

Separation of concerns through layering enables high cohesion and low coupling across the application components
Sample Reference Application Architecture Implementation Blueprint
(OMA/SOA Hybrid)

Various Third Party Products (as indicated)

Technology Products Mapped to the Reference App. Arch. Impl. Blueprint
(OMA/SOA Hybrid)

- Sample Product Mapping:
  - JEE Standard
  - IBM WebSphere Product Family
  - Various Third Party Products (as indicated)
## Agenda

<table>
<thead>
<tr>
<th>1</th>
<th>Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Architecture Blueprinting</td>
</tr>
<tr>
<td>3</td>
<td>Sample Architecture Blueprints</td>
</tr>
<tr>
<td>4</td>
<td>Architectural Mapping Process Illustrated</td>
</tr>
<tr>
<td>5</td>
<td>Conclusion</td>
</tr>
</tbody>
</table>

## Summary – Key Application Server Objectives

- Enable Rapid Development of Business Applications
- Provide Industry Leading Performance & Scalability
- Provide High Availability & Reliability
- Enable Enterprise Application Integration
- Allow Client-Independence (HTML, Java, C++, VB, etc.)
- Provide Open & Extensible Architecture
Class Project

- Project Description
  » The project focus is two-fold:
    • Based on a framework-based enterprise application of your choice, you will implement common facilities and application-level services on top of various types of application server technologies to support the various aspects of your chosen application
    • As you transition from one platform to another you will study and develop model-based migration and interoperability tools that leverage off of the cutting-edge concepts subsumed by modern Model Driven Architectures (MDAs)

Assignment

- Assignment:
  » #1a: Investigate legacy application server development environments (i.e., CGI-based). Write a short report that documents your findings and recommendations with respect to selection criteria in support of legacy development environments for application server technology
  » #1b: Come up with a framework-based business application and implement it on top of the core technology supplied as part of the homework #1 documentation
Assignments & Readings

- Readings
  - Handouts posted on the course web site
- Project Frameworks Setup (ongoing)
  - Apache Web Server (www.apache.org)
  - Perl

Next Session: Page-Based Application Servers