Implementation Architecture

Enterprise Application Architecture’s Forgotten Domain

Michael Hockridge
Vice President
(214)-777-4654
michael.hockridge@parivedasolutions.com

Pariveda Solutions, Inc.
Dallas, TX
Table of Contents

- Abstract

- Implementation Architecture Defined

- What happens when Implementation Architecture is absent?

- The three pillars of a successful Implementation Architecture

- Addressing a Deficient Implementation Architecture: Pariveda’s Approach
Abstract

- Implementation Architecture is a component of Enterprise Application Architecture that is critically important to the success of an organization's IT department.

- Implementation Architecture is often overlooked by today's Enterprise Architecture organizations and development teams are not equipped to manage the governance associated with a successful Implementation Architecture plan.

- Organizations without a strong Implementation Architecture will suffer from inefficiencies related to agility, quality and technical knowledge.

- A successful Implementation Architecture requires organizations to address three key competency areas.

- Companies can utilize Pariveda’s framework for Implementation Architecture to ensure their investments and activities yield maximum value for their particular organizations.
Implementation Architecture is one of six domains within Enterprise Application Architecture

- **Business Architecture**: involves aligning Business strategy with IT implementation, as well as, defining the organizational structure and governance.
- **Information Architecture**: provides a managed information environment for both operational and transactional data. It also enables the transformation of that data into information that supports analysis and reporting.
- **Application Architecture**: includes both conceptual and logical depictions of the application and focuses on commonality of services and application elements.
- **Technical Architecture**: describes the common infrastructure that supports all of the applications in the enterprise.
- **Implementation Architecture**: describes how to implement the application architecture on a specific technology.
- **Operational Architecture**: describes operational aspects such as monitoring, backup & restore, replication and deployment.

*Enterprise Architecture by Example - Cutter Consortium*
Here is an example of an organization with a poorly managed Implementation Architecture

<table>
<thead>
<tr>
<th>Key Focus Areas</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Lack of standards results in non-portable developers and difficulties transitioning to operations</td>
<td><strong>Applications</strong></td>
</tr>
<tr>
<td>2. Developers will frequently select new frameworks for personal reasons</td>
<td>.COM</td>
</tr>
<tr>
<td>3. Just because two applications utilize the same framework does not mean the underlying implementation is the same</td>
<td>Billing</td>
</tr>
<tr>
<td>4. Knowledge gaps on a particular development team will result in poor decisions</td>
<td>Portal</td>
</tr>
<tr>
<td>5. One-off selections can create unsupportable solutions</td>
<td>Help Desk</td>
</tr>
<tr>
<td>6. Lower level components are often the first to be standardized</td>
<td>Benefits</td>
</tr>
</tbody>
</table>

**Web**
- Content Delivery: GWT, Prototype, JQuery, Ext JS, JSP
- Layout: Tiles, Velocity, Tiles, Tiles
- MVC: Spring MVC, Struts2, Struts

**Integration**
- Web Services: Axis2, Spring WS, xfire
- Data Transform: AXIOM, JAXB, JAXB, Castor
- Guaranteed Deliver: Mule, RabbitMQ

**Batch**
- Scheduler: Quartz
- Job Config/Exec: Custom, Spring Batch

**Common**
- Persistence: Hibernate, Hibernate, Spring JDBC, Hibernate, JDBC
- Caching: ehcache, ehcache, JBoss cache, ehcache
- Logging: SLF4j, log4j, log4j, Commons, log4j

**Development**
- IDE: Eclipse, Netbeans, Eclipse, IntelliJ, Eclipse
- Build: Maven, ANT, Maven, ANT, ANT
- Config Mgmt: ClearCase, ClearCase, GIT, ClearCase, ClearCase

**Runtime**
- Web Server: IBM HTTP, IBM HTTP, IBM HTTP, IBM HTTP, IBM HTTP
- App Server: Websphere, Websphere, Websphere, Websphere, Websphere
Under investment in Implementation Architecture leads to inefficiencies in organizations and can be directly tied to negative outcomes for organizations.

**Inefficiencies**

**Agility**
- Limited to no reuse of architecture components
- Development is not prescriptive
- Developers are not portable
- SMEs are not portable
- Longer testing cycles
- Slow transition to Operations
- Development environment is not standardized

**Knowledge**
- Best practices are not implemented across the enterprise
- Lessons learned cannot be applied across the enterprise
- Enterprise training is not possible
- Developer boot camps are not possible

**Quality**
- Inconsistent Estimates
- Inconsistent Delivery

**Negative Outcomes**
- High Technical Debt
- Poor Performing Applications
- Delayed Realization of Business Value
- Additional Development Staff
- Additional Operations Staff
A successful Implementation Architecture requires organizations to address three competency areas:

**Methodology**
The Implementation Architecture methodology is comprised of the processes and artifacts that define the development patterns that are leveraged throughout the enterprise.

**Talent**
The following skill sets are required to develop an Implementation Architecture:
- Platform expertise
- Ability to operate within budgetary and delivery constraints
- Ability to exercise influence and judgment, while also demonstrating the ability to compromise

**Governance**
In order to ensure that the Methodology is consistently applied throughout the enterprise, organizations must have a strong Governance framework in place that places an emphasis on the accountability of architects throughout the development and delivery of an application.
An organization’s Implementation Architecture should align with its required Sophistication Level

**Sophistication Levels**

**Initializing**
- Arch Selection Guides
  - Frameworks
  - Tools
- Usage Documentation
  - Implementation patterns
  - Architecture usage guides
  - Coding standards
- Partial Platform knowledge
  - Limited implementation experience
  - Moderate understanding of key patterns and frameworks
- Consistent Compliance
  - Applications have some features in common with the architecture
  - The implementation of some features is not in line with the architecture

**Expanding**
- Training
  - Tutorials
  - Developer Boot Camps
- Community of Practice
  - Centralized repository that houses all artifacts
  - Accessible by the entire development community
- Full Platform knowledge
  - Moderate implementation experience
  - Full understanding of industry best practices
- Conformant Compliance
  - All aspects of the architecture are properly implemented
  - Some aspects are not covered by an architecture specification

**Formalizing**
- Development Machine
  - Preconfigured VM that contains core tooling
  - Runs the Reference App
- Reference Application
  - Working version of the arch
  - Demonstrates all patterns
- Ecosystem Knowledge
  - Extensive implementation experience
  - Awareness of bleeding edge trends
- Full Compliance
  - Full correspondence between an application and an architecture
  - The application fully adheres to the architecture and the architecture provides a specification for all aspects of the application
A well managed Implementation Architecture delivers value to the organization in the following ways

- IT organizations are better equipped to match the speed of the Business and can deliver value in a faster manner.

- Staffing reductions are possible for both Development and Operational teams given the gains in productivity.

- Applications will experience lower Technical Debt as a result of improvements to the quality delivered by development teams.

- Standards based development improves the effectiveness of development teams that operate in a distributed environment.

- Technical Subject Matter Experts (SMEs) are able to roam across projects and are not required to simply assist with the delivery of a single project.

- Improved application performance enables end users to be more efficient and provides them with an enhanced experience.

- Efficient usage of infrastructure is possible as a result of reduced system utilization.

- Applications will experience improved uptime metrics and Operational teams will be better equipped to manage production applications.
Pariveda can help clients by assessing the current state of their Implementation Architecture, recommending a future state and developing a road map for turning that plan into reality.

**Identify Current Sophistication Level**
- Pinpoint Implementation Architecture issues impacting Agility, Quality and Knowledge

**Determine Required Sophistication Level**
- Establish the end state vision for the Implementation Architecture

**Develop a Sophistication Improvement Plan**
- Plan the incremental steps required to address Agility, Quality and Knowledge gaps

<table>
<thead>
<tr>
<th>Identify Current Sophistication Level</th>
<th>Determine Required Sophistication Level</th>
<th>Develop a Sophistication Improvement Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determine the sources of technical debt within the enterprise</td>
<td>Identify and define the key Implementation Architecture</td>
<td>Develop a strategy for managing the Technical debt that has accrued in the enterprise</td>
</tr>
<tr>
<td>Analyze / document the tools and frameworks that are present in the enterprise</td>
<td>Identify the talent requirements needed to develop and deliver the Implementation Architecture</td>
<td>Determine quick wins that can return significant value with a limited investment</td>
</tr>
<tr>
<td>Analyze / document the various application architecture implementations</td>
<td>Define the processes required to develop and maintain an Implementation Architecture</td>
<td>Develop a Roadmap that prioritizes the realization of Implementation Architecture deliverables</td>
</tr>
<tr>
<td>Interview key resources in order to understand the design rationale behind the current implementations</td>
<td>Develop the process that assesses the compliance between the application and the architecture</td>
<td>Specify timelines to achieve architecture compliance goals</td>
</tr>
<tr>
<td>Assess the amount of compliance between the current architecture definition and the existing enterprise applications</td>
<td>Develop an accountability model that governs the interaction between architects and developers</td>
<td>Map planned initiatives to the Implementation Architecture delivery dates</td>
</tr>
</tbody>
</table>