Introduction to IT Architecture

&

Architectural Thinking
Think about the following questions on IT Architecture and the IT Architect’s role

- What is definition of IT Architecture?
- Why is IT Architecture important? Why do projects need it?
- What is the role of the IT Architect?
- Who is responsible for the solution’s architecture in a project?
- Who is responsible for the architectural documentation in a project?
- What is Architectural Thinking?
An architect is a person trained to plan and design buildings, and oversee their construction. To practice architecture means to provide services in connection with the design and construction of buildings and the space within the site surrounding the buildings, that have as their principal purpose human occupancy or use.

Architecture is both the process and product of planning, designing and construction.

Wikipedia Dictionary
What is an IT Architecture

IT Architecture defines the hardware and software structures of an IT System solution

• The components that will be reused, developed or purchased
• The externally visible properties of those components
• The relationships among components
  • Placement of components on nodes
  • Distribution geographically
  • Network connections and topology
  • How will they be managed
• How the components interact
  • Operational signatures
  • Interfaces
  • Protocols
Architecture provides a context and guidance, keeping everyone “on the same road”...
“A system is an entity which maintains its existence through the mutual interaction of its parts.” Austrian biologist Ludwig von Bertalanffy

Key concept is emergence (the whole is greater than the sum of its parts)

Note that a system exhibits properties that are not available in any of its parts.

**Architectural Thinking builds on Systems Thinking** by using a structured approach to deliver results
.. depends on the viewpoint and context to represent the architecture.
.. involves looking at the inputs, thinking process, and outputs.

- **Constraints** (What freedom we have?)
- **Requirements** (What is to be solved?)
- **Qualities** (How good should it be?)
- **Assets and Technology** (What is available?)
- **Strategy** (What we need / want?)

---

**Architectural Thinking**

- **Architecture**
  - **Design**
  - **Procurement**
  - **Development**
  - **Implementation**
  - **Operations**
Architecture’s all about creating and communicating good structure and behavior…
... with the intent of avoiding chaos...

... even if an individual house is well architected, if each house is different (e.g. different electricity voltage, water pressure) then the city will not work...
… with the intent of avoiding chaos...

… even if an individual house is well architected, if each house is different (e.g. different electricity voltage, water pressure) then the city will not work...
• Guarantee the system will support the users and the required business functionality

• Support system-wide requirements and qualities
  – Performance, availability, maintainability, portability, robustness, scalability

• Provide high-level structure for development effort
  – For design, buy and build
  – For work planning and allocation

• Allow for the development of reusable assets
What happens when we don’t?
There are many examples
There are many examples
There are many examples
As the size and complexity of systems increase, design and specification of the overall system becomes more important. The Architecture artifacts help to analyze competing concerns and to manifest them as systems qualities. They evolve from an ad hoc, word-of-mouth approach to an integrated set of documents. Below table lists only very commonly used artifacts;

- Architectural Overview Diagram
- System Context Diagram
- Architectural Decisions
- Functional Requirements
- Non-functional Requirements (NFR)
- Component Model
- Operational Model
• Description:
  – The Architecture Overview Diagram work product contains schematic diagrams that represent the governing ideas and candidate building blocks of an IT system. It provides an overview of the main conceptual elements and relationships.

• Purpose:
  — To communicate to stakeholders a conceptual understanding of the intended IT system
  — To provide a high-level shared vision of the architecture and scope of the proposed IT system for the development teams
What is System Context Diagram

• Usually shown as a diagram, this representation defines the system and identifies the information and control flows that cross the system boundary.
• It highlights several important characteristics of the system: users, external systems, batch inputs and outputs, and external devices.
• The objects within the system boundary define the scope over which the development team has some control.
Why did it fail?
• Architectural decisions capture precious knowledge that is worth sharing.

• Architectural decisions are significant because they may directly or indirectly determine whether a system meets its nonfunctional requirements.

• Architectural decisions, on the other hand, typically answer "why" questions, not just "what" and "how" questions.
### Example: Architectural Decision (AD)

<table>
<thead>
<tr>
<th>ID</th>
<th>Category</th>
<th>Decision Title</th>
<th>Question and Needs</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>AIX</td>
<td>User Logging Mechanism to Servers</td>
<td>How should users log on into the system?</td>
<td>ssh will be used. Telnet rlogin and rsh will be disabled.</td>
</tr>
<tr>
<td>002</td>
<td>AIX</td>
<td>Encryption for logging</td>
<td>How can we assure that the traffic between servers should not be sniffed and no user password shall be exposed?</td>
<td>SSL will be used for encryption.</td>
</tr>
<tr>
<td>003</td>
<td>AIX</td>
<td>AIX boot disk structure</td>
<td>Should every logical partition be booted from dedicated disks?</td>
<td>All logical partitions will have boot disks assigned from VIOS.</td>
</tr>
<tr>
<td>004</td>
<td>AIX</td>
<td>System p server database partition I/O allocation</td>
<td>Shall all I/O requirements be driven by VIOS?</td>
<td>Since database servers require high I/O demand, VIOS performance might be saturated by high I/O requirement. Therefore database servers will have dedicated host bus adapter and with external disk access without the use of VIOS.</td>
</tr>
<tr>
<td>005</td>
<td>AIX</td>
<td>VIOS Disk Availability</td>
<td>How should the VIO be implemented to ensure disk I/O high availability requirement?</td>
<td>Two VIOS will serve the same LUN. SDDPCM and MPIO will be used in the VIOS and in the LPARs.</td>
</tr>
<tr>
<td>006</td>
<td>AIX</td>
<td>CPU virtualization and free CPU utilization</td>
<td>The way that partitions use CPU must be defined. At System p CPU can be assigned to partitions either in dedicated or in shared mode – which is the preferred method?</td>
<td>Virtualization gives us an excellent opportunity to share the CPU and to utilize unused CPU cycles by different partitions that need for CPU power. Thus we need to use CPU in a shared mode.</td>
</tr>
<tr>
<td>007</td>
<td>AIX</td>
<td>CPU and priority setting</td>
<td>LPARs such as database servers should have priority over other servers in terms of CPU allocation. What will be the preferred mechanism to support it?</td>
<td>Weight will be defined each LPAR. CPU cycle operations will be done by the hypervisor based on the priority defined.</td>
</tr>
</tbody>
</table>
Why did it fail?
Functional Requirements:

- Are capabilities needed by users to fulfill their job
  Answers the question of "what" is wanted (but not "how" it is achieved)

Non-Functional Requirements:

Qualities:
- Define the expectations and characteristics that the system should support
  Might be runtime (for example, performance or availability) or non-runtime (for example, scalability or maintainability)

Constraints:
- Givens, those things that cannot be changed within the scope and lifetime of the project
- Other factors, such as mandated technologies, available skills, and budget
YOU'RE 10 MINUTES LATE, JIMMY!

IF ONLY JIMMY'D LEARNED ABOUT NONFUNCTIONAL REQUIREMENTS...

THIS THING'S FALLING APART! LOOK, THAT WHEEL'S COMING OFF.

JEEZ, LADY, IT'S GOT FOUR WHEELS AND IT GOES. WHAT MORE DO YOU WANT?

AND WHERE WERE YOU LAST SATURDAY WHEN I WANTED TO GO TO BECKY'S BIRTHDAY PARTY?
Possible Methods for documenting and Completing NFRs:
- Questionnaires;
- Document research;
- Interviews;
- Workshops.

Functional Requirements
- Direction wheel
- Moves forward

Non-Functional Requirements
- 2 seats
- Move 5Kmph

Future Requirements
- 1 baby seat
- Rainproof
Why did it fail?
Two major aspects of Architecture

• Functional aspect
  • Structure of software
  • Dynamic behavior (collaboration) of software "modules"
  • Defined through Component Model

• Operational aspect
  • Network topology (hardware nodes, locations, etc.)
  • What runs where (placement)
  • Service level characteristics (performance, availability)
  • Management and operation of IT System
  • Defined through Operational Model
Component Model – *notation*
*(structure described by relationship diagrams)*

```
component
+ DialogueControl
  + AccountProcessing
  + SecurityProcessing
  + CustomerProcessing
    - AccountMgr
    - SecurityMgr
    - CustomerMgr
```
Component Model – *notation*
*(behavior described by sequence diagrams)*
Why did it fail?
Operational model topology diagram

- For complex systems may need to divide into several pictures.
  - Network topology
  - Software services
  - Systems management
  - Location profiles

Supporting documentation
- Node descriptions
- Connection descriptions
- Node/deployment unit mapping
- Network description
- System management approach
- Middleware description
- Walkthrough descriptions
Why did it fail?
Why did it fail?
Why Architectures Fail

- Responses are based on an informal and unscientific survey
- All similarities between the responses and any real architectures or people are purely coincidental

Top Ten List

10. **Too complex** and unable to be managed and operated
9. **Bleeding edge** or unproven
8. Vendor promises of **future capability** are vaporware
7. No one understands the **entire** system
6. Not **scalable**
5. Evolves to which it was **never designed**, prototype modified
4. **Unclear**, missing **requirements**, or force fit
3. Resources over-challenged, **lack of skills** and experience
2. Lack of technical leadership and **governance**
1. **Management’s** fault
Questions !....