4-1 Architecture view

The Problem : Arch. document over-emphasize an aspect of development (i.e. team organization) or do not address the concerns of all stakeholders.

•The stakeholders of software system:

- end-user,
- developers,
- system engineers,
- project managers
- Software engineers struggled to represent more on one blueprint, and so architecture documents contains complex diagrams

Different stakeholders – different prospective

- Architecture also means different things to different stakeholders.
- Network Engineer would only be interested in the hardware and network configuration of the system;
- Project Manager in the key components to be developed and their timelines;
- Developer in classes that make up a component; and
- Tester in testable scenarios.

Solution

Solution came from “4+1” view model

- The views are used to describe the system from the viewpoint of different stakeholders, such as end-users, developers and project managers.
- Suitable for large and challenging architectures
- Describe different aspects of the system into different views.
- (Why?) Because different stakeholders have different aspects
- DEVELOPERS – Aspects of Systems like classes
- SYSTEM ADMINISTRATOR – Deployment, hardware and network configuration
- TESTER , PM, CUSTOMER -- ???
- “4+1” view model provides a “better organization with better separation of concern”.

4+1 View Model of Architecture

End user

Logical view

Development view

Programmers & software managers

Scenarios

Process View

Physical View

Integrator

System Engineer

4+1 view model architecture

1. Logical View (or Structural View) - an object model of the design
2. Process View (or Behavioral View) - concurrency and synchronization aspects
3. Development View (or Implementation View) - static organization (subset) of the software
4. Physical View (or Deployment View) - mapping of the software to the hardware
+1. Use-cases View (or Scenarios) - various usage scenarios

1.LogicalView = (The Object-oriented Decomposition)

- **Viewer:** End-user
- **Considers:** Functional Requirements- What the systemshould provide in terms of services to its users.

- **What this view shows?**
- “This view shows the components (objects)of the system as well as their interaction/relationship”.
- **Notation:** Object and dynamic models
- **UML diagrams:** Class, Object, State Machine, CompositeStructure diagrams

2. Process View = (The Process Decomposition)

- **Viewer: Integrator(s)**
  - **Considers:** Non-Functional Requirements - (concurrency, performance, scalability, usability, resilience, re-use, comprehensibility, economic and technology constraints, trade-offs, and cross-cutting concerns - like security and transaction management)
- **What this view shows?**
  - “This view shows the processes (workflow rules) of the system and how those processes communicate with each other”.
  - UML diagrams: Sequence, Communication, Activity, Timing, Interaction Overview diagrams

3. Development/Implementation View

= (The Subsystem Decomposition)

- **Viewer:** Programmers and Software Managers
  - Considers: Software module organization (Hierarchy of layers, software management, reuse, constraints of tools)
- **What this view shows?**
- “This view shows the building blocks of the system”.
- **UML diagrams:** Component, Package diagrams, Package Details, Execution Environments, Class Libraries, Layers, Sub-system design

4. Physical/Deployment View

= (Software to Hardware Mapping)

- Viewer: System Engineers
- Considers: Non-functional Requirements for hardware (Topology, Communication)

- What this view shows? Non-functional

- “This view shows the system execution environment”.
- UML diagrams: Deployment diagrams
- Non-UML diagrams: Network Topology (not in UML)

5. Use-case View/Scenarios = (putting it altogether)

- **Viewer:** All users of other views and Evaluators
- **Considers:** System consistency, validity
- What this view shows?
- “This view shows the Validation and Illustration of system completeness. This view is redundant with other views.”.
- **UML diagrams:** Use-case diagram, User stories

Relationships between Views

• The **Logical View and the Process View** are at a conceptual level and are used from analysis to design.

• **The Development View and the Deployment View** are at the physical level and represent the actual application components built and deployed.

• The Logical View and the Development View are tied closer to functionality (functional aspect) . They depict how functionality is modeled and implemented.
• **The Process View and Deployment View** realizes the non-functional aspects using behavioral and physical modeling.
• **Use Case View** leads to structural elements being analyzed in the Logical View and implemented in the Development View. The scenarios in the Use Case View are realized in the Process View and deployed in the Physical View.

Why is it called the 4 + 1 instead of just 5?

- The Use-case View: The use case view has a special significance. Views are effectively redundant (i.e. Views are interconnected).

However, all other views would not be possible without use case view.

It details the high levels requirements of the system.

The other views detail how those requirements are realized.

The Architecture must be Documented

Architectural Document should include (my version):

- Architectural goals & constraints
- Software Architectural Views:
  - Logical Views (representing user functionalities)
  - Process views (representing system execution)
  - Development Views (representing implementation breakdown)
  - Physical Views (representing deployment/hardware assignments)
  - Data view (representing the key files and tables)
- Scenarios
- Rationale
- Software Architecture (combined all the views and rationale) = {elements, forms, rationale}