Deployment/Operational Models

An Operational Model typically includes the following:

- A diagram showing the (candidate) groupings of hardware and software components that will comprise the overall system, and connectivity between those groupings.

- A description of each such candidate grouping, including its name, purpose (functions(s) within the system), and the components it contains.

- Several views of the system architecture, including Fit-for-Purpose Views reflecting the network topology, availability/scalability requirements, system security concept, a systems management arrangements, and a view of the development and test environments.

- An inventory of the business functions the system will support. A walkthrough of the Model, based on operational scenarios, should demonstrate how the operational groupings and their constituent components accomplish all required functions and business tasks.

An Operational Model can be used for different purposes at different stages of its development. Early on, the model serves as:

- A way of dividing large problems so that groups of functions can be worked on in relative isolation from one another
- A point of departure for early design reviews. Prior to selecting an implementation, the Operational model provides confirmation that the business problem has been well-articulated and that there is probably a viable IT solution. The model serves as a check that all the necessary business and technical functionality has been identified.
- The basis for early analysis of non-functional requirements such as system performance, availability, and capacity. The operational model specifies the desired non-functional characteristics of its components, and thereby helps to conduct trades among proposed solutions.

Later, the Model:

- Allows application developers to elaborate their designs based upon the model's representation of how the application will be implemented and managed
- Contributes to estimates of the cost of the infrastructure to be used -- both for budgeting purposes, and for building a business case for the solution

Note that it is risky to try to capture *all* operational aspects in a single diagram; such a diagram quickly becomes so complex that it is incomprehensible and ineffective. Architecture best-practice is to focus any given model or view within the architectural description on its key purpose, and to keep diagrams as simple as possible.

When it has been elaborated to the Specification level, an Operational Model can:

- Document the *grouping* of applications and technical subsystems ("deployment units") such that they can ultimately be installed on physical computer systems.
- Facilitate detailed design reviews
- Serve as a technical specification against which an architect can evaluate alternative hardware and software products (or even against which vendors can submit bids)
- Form the basis for a detailed prediction of performance, availability and other service level characteristics. (Note: Such predictions are based on the overall architectural description and the specifications of deployment units within it. They have to be revisited, via system tests, when specific implementations have been chosen).
- Allow application developers to refine and validate their architectural descriptions and designs, based on the Operational Model's detailed representation of all deployment units present in the solution.
- Allow cost estimation of the required infrastructure, for example using technology-neutral costing measures such as "dollars per megabyte of storage."

Once all relevant infrastructure information has been included in it, the Operational Model is considered to be at the "physical" level. Such a model can be used as a *blueprint* for the acquisition, installation and subsequent maintenance of the solution. It can also document how system elements at each location are managed, as well as the systems management components and nodes needed at each location.

Early (conceptual level) Operational Models form the basis for additional DoDAF-described Models and Fit-for-Purpose Views that add increasing detail and focus on specific operational aspects of the solution.

Later Models stage are capable of representing both COTS and custom-built technical components. In the case of custom-built applications, especially, deployment aspects can enter the trade space early in the process of architecting the system, while COTS apps may have fewer degrees of freedom with respect to deployment. Later in the architecting process, when a physical level of elaboration has been reached, deployment can be mapped at the level of physical groupings.

Mapping Deployment Units: the Deployment/Operational Model

A *Deployment/Operational Model* identifies "where" (by grouping) each component in the Component Model will be deployed. For each component, the concept of *deployment* has four possible aspects:

- Installation – within which grouping(s) will the component be installed?
- Execution –within which grouping(s) will the component perform its function(s)?
- Data (state) – from which grouping(s) will the component receive data, and to which ones will it pass its own data?
- Presentation (i.e., user interface) – to which grouping(s) will the component have a presentation interface?

Depending on how the groupings have been configured within the architectural description, all four of the aspects above may be mapped to one grouping -- for example, if the component is a standalone desktop application. In other cases, a component may be architecturally associated with (mapped to) one grouping with respect to Installation, to another for Presentation, and to several others with respect to Data (e.g., a distributed web application).

The collection of components that have a deployment aspect on a particular grouping are referred to as a *deployment unit*. For example, a collection of Java classes installed on a specific Server grouping would be a deployment unit. Component deployment can be visually represented on Operational Model diagrams by annotating groupings with a list of abbreviated component identifiers and the deployment aspect that applies to that grouping/component combination. Deployment details can also be captured in a Deployment Matrix Table that maps components to groupings.