Architecture-Based Systems Engineering and Integration

By
Dr. A.R. Habayeb
Virginia Polytechnic Institute & State University
E-mail: ahabayeb@vt.edu
8th Annual Systems Engineering Conference Oct. 24-27, 2005
San Diego, CA
Outline

1.0 Why Architecture-Based Systems Engineering?  
   What does Architecting bring to the Systems  
   Engineering Process?
2.0 DODAF views and Products
3.0 Four Aspects of Systems Engineering.  
   Five Phases of Systems Engineering Process.
4.0 Five System Architectures;  
   The 4+1 Software Architectures.
5.0 Merging Activities: Architecting, Engineering  
   Integration, and Evaluation.
6.0 System Integration and Interfaces.
7.0 Conclusion
Why Architecture-Based Systems Engineering?

- DOD Systems Acquisition paradigm is moving away from acquiring products to acquiring Capabilities.
- Emphasis on using existing systems with new systems.
- Network Centric Operations demands flexibility and Interoperability between System-of-Systems.
- Coping with Complexity, uncertainty and cost of Engineering Effective Systems and Enterprises.
- Utilizing Information Technology to improve Productivity, integration, and build unprecedented Complex Systems.
Architecture-Based Systems Engineering

- Responding to these changes and challenges: We must systematically merge: Architecting activities, Engineering, Integration, and Evaluation of systems.
- The response is using the Systems Engineering approach of iterative top-down design and bottom-up integration of software and hardware.
- DOD has directed that DODAF be used to describe several Aspects of system architecture. DODAF consists of 26 views products.
- DODAF 1.0 does not provide guidance for detailed design, integration, or evaluation of systems.
- DODAF products can be used to create the System Architecture.
Architecture-Based Systems Engineering

ABSE Flow

Architecting Process

Standards
*Visualization
*Modeling: Structured Analysis; Object Oriented
*Diagramming: IDEF0, IDEF1x, FFBD, EFFBD

Requirements (Raw)

*System Architectures
*System Level Specification

Requirements (Defined)

Systems Engineering Process

*Responsive Systems
*Performance & Design Specification
*Certification

Tools

*Systems Analysis
*Development Models
*Simulation
*Modeling
*Test and Evaluation
Concept of Architecture

- The concept of architecture entered the domains of Software and Systems Engineering in recent decades.

- Some people argue that the Software and System architecture are similar to that of civil architecture.

- A building architecture is a detailed endeavor driven by the user that is meant to stay for many years. The drawings are very detailed to support the builder.

- Enterprise/System architecture must be more responsive to change & extensible: flexible, adaptable, and scalable than building architecture. It must be interchangeable / modular, & reusable.
Types of Architectures

There are two types of architectures:

1. **System Architecture** of building real systems: Civil, Hardware, and Embedded systems architecture,

2. **Domain Specific Architectures**: Software, supply chain, finance, insurance, business processes, security, C4ISR, avionics, weapon system.

** Architectures are preambles and part of the systems engineering process.
Architecture Framework

- **Framework** is the structural frame of reference,
- Architecture Framework e.g. DODAF or Zachman give the structure of the architecture.
- **Framework does not give** detailed guidance on how to design or implement architectures.
- **Architecture (Framework +Design) = System Architecture**
DODAF Linkages Among Views

Figure ES-1. Linkages Among Views
System Architecture Must Have:

- Every system/enterprise must have an architecture
- An architecture must have:
  * Structure (Topology)/Organization,
  * Functionalities: Function Blocks, Modules
  * Connectivity: Linkages and interfaces-
    Function Blocks Diagram for a given set of events
Architecture-Based System Design Process

• Two approaches:
  (1) Structured Analysis (SA):
      using IDEF0, IDEF1x, DFD, FFBD, EFFBD
  (2) Object Oriented (OO):
      using UML or equivalent
Systems Engineering

Systems engineering is a multidisciplinary subject dealing with the integration of all parts of a system (hardware, software, and operator) into the real world environment. It provides rationalization for tradeoffs in meeting the requirements and building the system.
Systems Engineering Framework

Philosophical
- Theory
- Concepts
- Process: phases, steps, stages
- Development Models

Management & Business
- Planning, budgeting
- Costing
- Schedule control
- Configuration Control
- Program management
- System acquisition
- Political engineering

Technical & Engineering
- Effectiveness factors
- SEA
- Metrics
- Capability
- Performance
- Testing
- Reliability
- Maintainability

Operational
- Derivation of requirements
- Validation of requirements
- Testing
- Logistics
- Support
- Training
Systems Engineering Process

Systems Engineering Process deals with all aspects of system design, development, integration, evaluation, disposal, and management.

(1) Determination of requirements,
(2) Definition of system architectures,
(3) Establishing performance, reliability, and availability goals
(4) Tradeoff analysis: performance versus cost, reliability, support,
(5) Decomposition and partitioning of system, and system design,
(6) Integration of building blocks, interfaces, testing, and safety,
(7) Metrics to evaluate all these activities, evaluate the effectiveness of alternatives and options,
(8) System engineering management: system acquisition, project management, and risk analysis.
Partitioning Levels of System Architecture

There are five levels of architectures:

1. **System/Enterprise** level → Conceptual Architecture
   Expressing stakeholders needs, concerns, capability requirements, and strategy

2. **Subsystem** level → Functional/Logical Architecture
   Performance, functional connectivity, and interfaces

3. **Module** level → Physical Architecture
   Physical connectivity and interfaces

4. **Function** Block level → Operational/Implementation Architecture

5. **Subfunction** level → Bottom-up Integration and Interfaces Architecture

* Physical and functional connectivity are interfaces driven
** Functional and Physical architectures are performed concurrently
** The final architecture is achieved by iteration between these levels
** These architectures evolved into architecture Views: System, Operational, Technical and All
System Architectures Flow and Relationship

Conceptual Architecture
Concept of Operation

Needs and Requirements
Operational Concept

Functional/Logical Architecture

Physical/Structural Architecture

Integration and Interfaces Architecture

Operational Architecture

Architecture flow development in the engineering of a system
System Architecture Model

The relationship of the five kinds of system architecture:

- Conceptual
- Physical
- Integration & Interfaces
- Functional
- Operational

Stakeholders, End-users, & Analysts → Conceptual
Architects, Designers, Developers → Physical
Architects, Designers, Developers → Integration & Interfaces
Evaluators, Operators, Integrators → Functional
System: Integrators, Designers, Operators → Operational
Conceptual Architecture Development Process

- Develop Operational Concept
  - Define System Boundaries and External Interfaces
  - Design Changes
  - DODAF: AV-1, OV1, OV-2, OV3, SV-1
  - Requirements Changes, Proof-of-Concept experiments

- Operational Architecture

- Develop Feasible Requirement
  - System Analysis
    - AOA
    - COEA
    - Cost-Effectiveness
    - Proof-of-Concept

- Conceptual Architecture, Scenario

- System Requirements

Stakeholders:
- Requirements
- Needs
- Concerns
- Scenarios

Iteration between the development levels
Functional Architecture Development Process

Conceptual Architecture And System Requirements

Operational Concept

Generate Functionalities From Operational Concept

Functionalities List

Determine the Relationship among Inputs/Outputs

Establish Data Model

Input/Output Requirement

Boundary Inputs, Control And Objectives

Functional Decomposition Using IDEF0 (Integrated Definition for Function Modeling)

DODAF: AV-1, OV-1,2,3, SV-1,2,4, TV-1
Physical/Structural Architecture

• The physical architecture gives the physical resources to perform the system functions. It is developed concurrently with the Functional Architecture.

• It identifies resources to form the structural architecture. Physics-Based simulations provide insights into the physical architecture.

• Physical architecture is a description of the partitioned elements of the system without their performance specifications.

• It accounts for all the nonfunctional attributes: Reliability, availability, security, scalability, reusability.

![Diagram of Physical/Structural Architecture]
Physical Architecture Development Process

1. **Conceptual Architecture**
   - System Level Functional Architecture
   - **Match Functions to Resources**
   - Identify Resources For every function

2. **Framework Physical Architecture**
   - DODAF: OV-1,2,7
   - SV-1,4,7; TV-1,2

3. **Apply Performance Specification to the resources**

4. **Select Physical Architecture**

5. **System Requirements**
   - System Boundary Objective
   - *Tradeoffs Analysis*
   - Alternatives Versus KPPs Matrix

6. **Physical Architecture Changes**

   - Physical Architecture Block Diagram
Physical Architecture of a Weapon System

Work Breakdown Structure is similar to the physical architecture (WBS)
Reliability/Availability Block Diagram

Example of a Physical/Structural architecture

Series Architecture

Series parallel Architecture
Development Process of Integration and Interfaces Architecture

Functional Architecture

Conceptual Architecture; Stakeholders Requirements

Tradeoffs Analysis

Physical System Partitioning

Operational System Partitioning

External Connectivity

Functional System Partitioning

DODAF: OV-1,2,3,6a, SV-1,2,3,4,6,TV-1

Co-design of Embedded Software

Physical Architecture

Changes to Mission & Objective

Operational Interfaces

Design Specification

Prototype

Interoperability Issues
Development of Operational Architecture

Allocate Functions To Physical subsystems

Function to Subsystem Allocation

System Analysis
Tradeoffs Analysis
Risk Analysis
Operational Analysis

Subsystem Performance Specification

External Connectivity

Integration and Interfaces Architecture

DODAF: AV1, OV-1,2,3,4,5,6a
SV-1,2,

Operational Concept, Use cases

Requirements
Conceptual Architecture

Physical Architecture

Functional Architecture
System Level Specifications

Alternative Operational Architecture

Operational Architecture

ARH 25
Systems Engineering Process
Seven Phases for Acquisition/Production

Definition Phases
- Requirements Specification
- Conceptual Design

Design & Development Phases
- Logical Design & Architecting
- Detailed Design & Testing
- Operational Implementation

Deployment Phases
- Conceptual Architecture
- Functional Architecture
- Physical Architecture
- Integration and Interfaces Architecture
- Operational Architecture
- Operational Test & Evaluation
- Operation and Maintenance
First prototype
Is a proof of
Concept.
2nd prototype
Accounts for the
Architecting
process.
Understand User Requirements, Develop System Concept and Validation Plan

Develop System Performance Specification and System Validation Plan

Expand Performance Specifications into CI “Design-to” Specifications and CI Verification Plan

Evolve “Design-to” Specification into “Build-to” Documentation and Inspection

Inspect to “Build-to” Documentation

Fabricate, Assemble and Code to “Build-to” Documentation

Assemble CIs and Perform CI Verification to CI “Design-to” Specifications

Integrate System and Perform System Verification to Performance Specification

Demonstrate and Validate System to User Validation Plan

Integrate System and Perform System Verification to Performance Specification

Systems Engineering

Design Engineering

Operational Architecture

Integration and Qualification

Physical Architecture

Functional Architecture

Conceptual Architecture

Decomposition and Definition

Integration & Interfaces Architecture

CI-Configuration Item

Time

ARH 28
Systems Engineering Interactive Model

**System Definition**
- Requirements/Objectives
- Requirements Analysis
- Mission Analysis
- System Performance
- Preliminary Conceptual Design

**Requirements Loop**

**System Development**
- System Effectiveness Analysis
- MOEs, MOPs
- Functional Requirements
- Tradeoffs Analysis
- Technology Base
- System Design and Architecting
- System Integration

**Design Loop**

**System Deployment**
- Detailed Design and Testing
- Transform System Architecture into Real World Entities: Functional and Physical
- System Design Specifications and Standards
- Operational Implementation
- Operational Test and Evaluation
- Operation and Maintenance
Software Architecture

- Software architecture deals with the design and implementation of the high-level structure of software.
- A model of five views called (4+1) is used to describe the software architecture:
  1. Logical View,
  2. Process View,
  3. Development View,
  4. Physical View
  5. Use Case view
Software Architecture Model

Four + One View Model

End-user
Functionality

Evaluation
Verification
Testing

Programmers
Software Management

Integrators
Performance
Scalability

Logical View

Use Cases
Scenarios

Development
View

Processes
View

Physical View

Systems
Engineers
Communications
Topology
Merging Architecting Into the Systems Engineering Process

<table>
<thead>
<tr>
<th>Partitioning Level</th>
<th>Software Architecture Views</th>
<th>Systems Engineering Process: 5Ds</th>
<th>System Architecture Views</th>
<th>DoDAF: C4ISR Views</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>Logical Architecture</td>
<td>Definition Phase</td>
<td>Conceptual Architecture</td>
<td>ALL View</td>
</tr>
<tr>
<td>Subsystem Cluster</td>
<td>Process Architecture</td>
<td>Design Phase</td>
<td>Functional Architecture</td>
<td>Operational View</td>
</tr>
<tr>
<td>Module</td>
<td>Development Architecture</td>
<td>Development Phase</td>
<td>Physical Architecture</td>
<td>Systems View</td>
</tr>
<tr>
<td>Function Block</td>
<td>Physical Architecture</td>
<td>Deployment Phase</td>
<td>Integration &amp; Interfaces Arch.</td>
<td>Technical View</td>
</tr>
<tr>
<td>Subfunction</td>
<td>Use Cases Architecture</td>
<td>Disposal Phase</td>
<td>Operational Architecture</td>
<td></td>
</tr>
<tr>
<td>Component Element</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Architect and Systems Engineer Spaces

[Diagram showing the relationship between Architecture Problem Space and Solution Space Engineering]
Level of Architecting Involvement in the Systems Engineering Process

![Diagram showing level of involvement in various stages of the systems engineering process.]

- **Concept Exploration**: A (Architect) and SE (Systems Engineer)
- **Preliminary Design**: A (Architect) and SE (Systems Engineer)
- **Detailed Design**: A (Architect)
- **Integration & Interfaces**: SE (Systems Engineer) and SE (Systems Engineer)

Legend:
- A: Architect
- SE: Systems Engineer
System Integration and Interfaces: Why is it important?

- System Integration is a critical part of the Systems Engineering Process, it brings the decompositions together.

- Interfaces are the keys to a successful Integration.

- The recognition and identification of Interfaces are very important in building responsive, robust, and Effective systems.

- Interfaces occur at the boundaries of the building Blocks of the system: functionally, physically, and operationally.

- The capability of recognizing and counting the interfaces will ensure the success of the architecting and engineering process.
The Binary Interfaces Model

- We build systems to give outcomes. System behavior is described by its outcomes.
- We will use a binary building blocks model to determine and count interfaces.
- Given a system whose functionality is achieved by integrating (merging) $n = \text{three building blocks: A, B, C}$. 
- The number of possible outcomes is $2^n = 2^3 = 8$.
- The outcomes are represented by:
  1. Events (Euler, or logical) Diagrams,
  2. Network (Decision tree),
  3. Truth table,
Binary Model of Interfaces

- When the functionalities of A, B, and C merge new functions are created and are represented by the events (logic) diagram.

Interfaces occur when two or more functionalities are merged, we call them mathematical intersections of order two or more. For this example:

There are three second order interfaces: \( A \cap B \cap \overline{C} \), \( A \cap \overline{B} \cap C \), \( \overline{A} \cap B \cap C \)

and a third order interface: \( A \cap B \cap C \)
Subsumed Interfaces

• As we integrate systems not all interfaces are recognized and accounted for.
• The question is what happens when interfaces are subsumed?
• Subsumed interfaces are either absorbed by other interfaces, or they are neglected. This can happen when functionalities are not merged properly, poor coupling, low amplification, or by organizational design.
• The impact of subsumed interfaces is serious degradation in system effectiveness.
Logically Integrated System versus System with Subsumed Interfaces

The area of the function is proportional to its: **Design Limitation=1-Design Adequacy**

System S1: 2 Subsumed Interfaces

- **Design Limitation** = A+B+C-AB-AC
- **Design Adequacy** = 1-A-B-C+AB+AC

System S2: Logically Integrated

- **Design Limitation** = A+B+C-AB-AC-BC+ABC
- **Design Adequacy** = 1-A-B-C+AB+AC+BC-ABC

<table>
<thead>
<tr>
<th>A=B=C</th>
<th>Adequacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>.1</td>
<td>.72</td>
</tr>
<tr>
<td>.2</td>
<td>.48</td>
</tr>
<tr>
<td>.3</td>
<td>.28</td>
</tr>
<tr>
<td>.4</td>
<td>.12</td>
</tr>
<tr>
<td>.5</td>
<td>0</td>
</tr>
<tr>
<td>.7</td>
<td>0</td>
</tr>
<tr>
<td>1.0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A=B=C</th>
<th>Adequacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>.1</td>
<td>.729</td>
</tr>
<tr>
<td>.2</td>
<td>.512</td>
</tr>
<tr>
<td>.3</td>
<td>.343</td>
</tr>
<tr>
<td>.4</td>
<td>.216</td>
</tr>
<tr>
<td>.5</td>
<td>.125</td>
</tr>
<tr>
<td>.7</td>
<td>.027</td>
</tr>
<tr>
<td>1.0</td>
<td>0</td>
</tr>
</tbody>
</table>
Comparing Two Systems

System $S_1$ with subsumed Interfaces       System $S_2$ Logically Integrated

PARAMETER LIMITATION

$P_{da}$
Conclusion

- Architecting is a major contributor to the engineering of responsive systems. It strengthens the systems engineering process and provides critical insights into the stakeholder and user requirements.

- System architecture and software architecture should be congruent.

  * Interfaces are the keystones of system integration.

  * Merging the architecting, engineering, integration, and evaluation into the Systems Engineering process will provide a comprehensive and a balanced approach to deal with the complexity of building effective systems.
Architecture-Based Systems Engineering and Integration

Backup
Next Generation Systems

- Traditional Systems Engineering and Systems Acquisition are focused on Vertical (Stovepipe) System design.
- System acquisition have moved away from the traditional model to acquiring Capabilities.
- Next generation systems must be flexible, scalable, interoperable, and adaptable/agile. Therefore, the design of Future Systems should be based on “Capability Building Blocks”
- Current and Future Network Centric Operations requirements mandate Interoperability.
- Current and future systems are software intensive. Software is embedded into systems and intermingled within System-of-Systems.

** We need a comprehensive approach to build systems, whose architectures are compatible with the systems engineering process.
System Architecture Model

The system architecture model is the model of function and form that defines the physical structure, functional flow, interfaces, and the connectivity of the system.
Architecture Implementation

- **Architecture Implementation Principles**
- Architecture framework
  - DoDAF
  - Zachman
- Implementation methodology of software
  - SOA
  - Data driven
  - Client/Server
  - **Model driven architecture (MDA) using UML**
  - Event driven architecture (EDA)
  - **Component-based architecture**
  - Aspect Driven Architecture
  - Domain Driven Architecture
## Input/Output of System Design Activities During Architecting and Engineering

<table>
<thead>
<tr>
<th>Design Activity</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Conceptual System Design</td>
<td>Stakeholders Needs &amp; Concerns Operational Concept</td>
<td>Requirements Conceptual Architecture Concept of Operation</td>
</tr>
<tr>
<td>(4) Integration &amp; Interfaces</td>
<td>Functional &amp; Physical Architecture Performance Specification</td>
<td>Integration &amp; Interfaces Architecture Design Specification KPPs</td>
</tr>
<tr>
<td>Subsystem, Module, Function Block</td>
<td></td>
<td></td>
</tr>
<tr>
<td>And Risk analysis</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(6) Evaluation and Testing</td>
<td>Integration &amp; Interfaces &amp; Operational Architecture</td>
<td>Design Parameters Tradeoff &amp;Feedback</td>
</tr>
<tr>
<td>of system design</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conceptual Architecture

- The *conceptual architecture* is a **high level abstraction of the requirements**. Virtual Simulation provides insights into the conceptual architecture.
- It is obtained by partitioning the requirements based on the **stakeholders needs, concerns, and requirements**.
- The conceptual architecture establishes the system/enterprise **design requirements** based on: Input/Output, Performance and cost tradeoffs, and mission analysis.
- **Proof-of-concept** experiments are used to support the Conceptual Architecture.

* In software engineering conceptual architecture can be derived from “use cases”, scenarios, and activity diagrams.*
Functional/Logical Architecture

- The **functional architecture** contains the system functions, configuration units, building blocks and components. It is represented by a **directed network**.

- It is a logical representation of **what the system must do?**, it provides a decomposition of the system objective. **Physics-based simulations** give insights into the functional architecture.

- A logical model of **transforming inputs** into **outputs** using control information-flow throughout the functional decomposition. It defines the **functions and the data flows**.

- It is derived using an **IDEF0 model, FFBD, DFD, or “OO” UML methodology**.

** Functions express activities, actions
*** Form follows function, Physical follows functional
*** Fit follows form which follows function: **Function, Form, Fit**
Functional Decomposition

- Several system features are used to partition a system into building blocks
  1. At the top level systems operate in modes. Modes lead to partition the system into subsystems.
  2. Each subsystem is partitioned into modules based on grouping of functions.
  3. Modules that have multiple inputs and outputs are partitioned by tracing the input to the output to establish the function.
  4. Partition each function block into subfunctions: Circuits/Applications.
  5. The lowest level of the functional architecture identifies devices and activities such as, transmit output, receive input, store output, format input, amplify input.
The **operational view** provides the information exchanges, interoperability levels, and performance parameters required to support a mission or task.

The **systems view** defines system attributes, and provides the basis for comparing system performance against operational requirements.

The **technical view** defines the **standards** to the implementation criteria for fielding an interoperable system.

The three views and their interrelationships are designed for deriving measures such as interoperability, performance and measuring the impact of these metrics on operational mission and task effectiveness.

Many data elements of the products are used in more than one product and there are several mapping relationships between products.
Integration and Interfaces Architecture

- **Integration is a** bottom-up process of assembling the system from its components. Engineering-Based simulations and mockups provide insights into the integration and interfaces architecture.

- Integration is a hierarchical process. At the top level, it is the merging of subsystems functionally and physically. At the subsystem level it is the merging of modules functionally and structurally.

- Integration and Interfaces Architecture merges the functional, physical, and conceptual architectures.

- **Interfaces** are critical parts of the systems.

- **Interoperability** is based on system-to-system interfaces; it is a major driver in architecting SoS& FoS.
Operational Architecture

- The operational architecture brings together through scenarios the requirements decomposition (conceptual architecture) with the functional, physical, and integration and interfaces architectures.

- Scenarios are used to develop the operational architecture. It identifies the system’s internal and external interfaces, connectivity, and nodes.

- It provides sufficient details to evaluate system performance, performance tradeoff analysis, and risk analysis.

- Operational architecture describes the mission, tasks, operational elements, and information flows required to accomplish or support system functions.

- It is concerned with interoperability and human interfaces. It defines and analyzes functional activation and control structure. Engineering-Based Simulation provides insights into the operational architecture.
Integration and Interfaces Architecture Continue

- Interfaces are connections for tying the system parts to each other, and creating new functions and nodes.

- There are internal interfaces of merging one system part to another. External interface is the connection of the system to another system-Interoperability.

- Integration and interfaces are united activities in building systems.

- Interfaces are the basic building blocks of the system integration process.

- There are several types of interfaces: Electrical, logical, physical, environmental, communications, thermal, social (human interfaces).

- Interfaces are the mathematical intersections of the interacting functions that create new functions, or activities.
Products of Architecture Views

• Each view is described by a set of products, e.g., graphics, diagrams, tables, scripts.
• Each product contains “data” that describes some aspect of the architecture, e.g., functionality, structure, connectivity.

Issues:
(1) Are the products consistent?
(2) What are the relationships among the products?
(3) Does the group of products contain all the information that define an architecture?
Products of Architecting

- Architecting is a key upfront activity and is strongly tied to the customer and user (stakeholder).
- Architecting is a major part of the definition and design phase of the Systems Engineering Process
- **Architecting Products To Systems Engineering**
  1. Provide visualization of needed capability based on requirements
  2. System Conceptualization and Innovation
  3. Interfacing with stakeholders
  4. Provide Architecture Framework: DODAF
  5. Generate and balance operational views and systems views
  6. Identify Standards
  7. Provide capabilities road-map
  8. System partitioning: functional, Physical, Logical, and Operational
  9. Define System boundaries
  10. Identify tradeoffs space: Cost, Capability, Support, Reliability, availability, and Risk
  11. Identification of interfaces (external and internal)
  12. Modeling/operation analysis
Products of Systems Engineering

System Definition, Design, Development/Production, Deployment, and Disposal.
(1) Visualization of what a system should look like: Engineering the System
(2) Tradeoff studies – AOA - CE, CB, Risk
(3) Requirements decomposition/Functional Decomposition, traceability
(4) System and design specification, and Configuration Management
(5) Detailed system analysis and identification of capability drivers
(6) Detailed system partitioning
(7) Detailed System integration and interfaces; codesign issues
(8) System Acquisition
(9) Resource allocation (scope)
(10) Integrated Logistics Support (ILS)
(11) Select metrics: MOEs, MOPs
(12) Data exchange and collaboration
(13) Test and Evaluation Plan
(14) Management of the Systems Engineering Process
Systems Engineering Attributes

- Systems Engineering is the integrating mechanism across the technical efforts related to the design, development, acquisition, integration, manufacturing, verification, deployment, operations, support, disposal, and user training of systems and their life cycle processes.
- Systems Engineering develops technical information to support the program management decision-making process in meeting: Cost, Schedule, Performance and Risk.
## “ALL Views”
### Two Views

<table>
<thead>
<tr>
<th>Framework View</th>
<th>Product Name</th>
<th>General Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AV-1</td>
<td>Overview and Summary Information</td>
<td>Purpose, scope, stakeholders, user, environment description, analytical assessment</td>
</tr>
<tr>
<td>AV-2</td>
<td>Integrated Dictionary</td>
<td>Data repository with definitions of terms used in all products</td>
</tr>
</tbody>
</table>
## Operational View

### Nine Views

<table>
<thead>
<tr>
<th>Framework Product</th>
<th>Product Name</th>
<th>General Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OV-1</td>
<td>Graphic high Level Operational Concept</td>
<td>High-level graphical &amp; textual description of operational concept</td>
</tr>
<tr>
<td>OV-2</td>
<td>Operational node Connectivity Description</td>
<td>Operational nodes, operational activities at each node, connectivity and information exchange need-lines between nodes</td>
</tr>
<tr>
<td>OV-3</td>
<td>Operational Information Exchange Matrix</td>
<td>Information exchanged between nodes, and the relevant attributes of that exchange</td>
</tr>
<tr>
<td>OV-4</td>
<td>Organizational Relationship Chart</td>
<td>Operational role, or other relationships among organizations</td>
</tr>
<tr>
<td>OV-5</td>
<td>Operational Activity Model</td>
<td>Capabilities, operational activities, relationships among activities, inputs, and outputs. May show cost.</td>
</tr>
<tr>
<td>OV-6a</td>
<td>Operational Rules Model</td>
<td>Describes sequence and timing of operational activities - identifies business rules that constrain operation</td>
</tr>
<tr>
<td>OV-6b</td>
<td>Operational State Transition Description</td>
<td>Describes sequence and timing of operational activities identifies business process response to events</td>
</tr>
<tr>
<td>OV-6c</td>
<td>Operational Event Trace Description</td>
<td>Describes sequence and timing of operational activities traces actions in a scenario or sequence of events</td>
</tr>
<tr>
<td>OV-7</td>
<td>Logical Data Model</td>
<td>Data requirements documentation and structural business process rules of the Operational View</td>
</tr>
<tr>
<td>Framework Product</td>
<td>Product Name</td>
<td>General Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SV-1</td>
<td>Systems Interface Description</td>
<td>Identification of systems nodes, systems, items, and their interconnections within and between nodes</td>
</tr>
<tr>
<td>SV-2</td>
<td>Description of Systems Communications</td>
<td>Systems nodes, systems, and system items and their related communications lay-downs</td>
</tr>
<tr>
<td>SV-3</td>
<td>Systems-Systems Matrix</td>
<td>Relationships among systems, system-type interfaces Planned vs. existing interfaces</td>
</tr>
<tr>
<td>SV-4</td>
<td>Systems Functionality Description</td>
<td>Systems Functions and data flow among them</td>
</tr>
<tr>
<td>SV-5</td>
<td>Operational Activity to Systems Function Traceability Matrix</td>
<td>Mapping systems to capabilities, functions and Operational activities</td>
</tr>
<tr>
<td>SV-6</td>
<td>Systems Data Exchange Matrix</td>
<td>Systems data elements exchanged between systems and the attributes of that exchange</td>
</tr>
<tr>
<td>SV-7</td>
<td>Systems Performance Parameters Matrix</td>
<td>Performance characteristics of elements for a given timeframes)</td>
</tr>
</tbody>
</table>
## Systems View
### Six Views

<table>
<thead>
<tr>
<th>Framework Product</th>
<th>View Name</th>
<th>General Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SV-8</strong></td>
<td>Systems Evolution Description</td>
<td>Planned migration of systems to more efficient suite, or Evolving a current system to future implementation</td>
</tr>
<tr>
<td><strong>SV-9</strong></td>
<td>Systems Technology Forecast</td>
<td>Time table of emerging software &amp; hardware products Technologies and that will impact future architecture</td>
</tr>
<tr>
<td><strong>SV-10a</strong></td>
<td>Systems Rules Model</td>
<td>Describes systems functionality constraints due to some aspect of systems design or implementation</td>
</tr>
<tr>
<td><strong>SV-10b</strong></td>
<td>Systems State Transition Description</td>
<td>Describes systems functionality: identifies responses of a system to events</td>
</tr>
<tr>
<td><strong>SV-10c</strong></td>
<td>Systems Events Trace Description</td>
<td>Describes systems functionality: identifies system changes due to critical sequences of events of the Operational View</td>
</tr>
<tr>
<td><strong>SV-11</strong></td>
<td>Physical Schema</td>
<td>Physical implementation of the Logical Data Model entities: message formats, file structure, physical schema</td>
</tr>
</tbody>
</table>
## Technical Standards View Products

### Two Products

<table>
<thead>
<tr>
<th>Framework Product</th>
<th>Product Name</th>
<th>General Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV-1</td>
<td>Technical Standards Profile</td>
<td>Listing of standards that apply to System View elements in a given architecture</td>
</tr>
<tr>
<td>TV-2</td>
<td>Technical Standards Forecast</td>
<td>Description of emerging standards and potential impact on current System View Elements within a set of timeframes</td>
</tr>
</tbody>
</table>
Tailoring the Views

• Not all software architectures need all five views. **Not relevant views** can be omitted from the architecture description. For example, the physical view can be omitted if there is only **one processor**, also, the process view can be omitted if there is only **one program**. For very small system it is possible to merge the logical with the development view.

• Similarly, in the DODAF not all views are essential. In general, **AV-1, AV-2, Ov-1, OV-2, OV-3, SV-1, and SV-4** are essential. The remaining products are supporting views.
Logical Software Architecture & Process Software Architecture

- The Logical Architecture supports the functional requirements; it describes the “Object Oriented” system decomposition using: Class, Collaboration diagrams and the sequence diagram.

- The Process Architecture gives Process partitioning into Tasks; it describes how processes communicate and interact. It accounts for the nonfunctional attributes: performance, availability, reliability, security, and scalability. Activity diagrams are used to describe this view.
Development Software Architecture

- The Development Architecture describes the actual software modules of the system and its interaction with the environment. Packages, subsystems, and class libraries are considered modules.
- The Development view gives the layers of the system:
  1. User Interface layer,
  2. Presentation layer,
  3. Application Logic,

* The Development Architecture is given by module and subsystem diagrams.
Physical Software Architecture

- The Physical Architecture is a mapping of Software onto the Hardware. It describes how the applications are installed into the hardware, and how it executes in a network environment. It takes into account the non-functional requirements of the system: reliability, performance, availability, reuse, scalability. Deployment diagrams are used to represent the Physical Architecture.
The “Plus One” View
Use Case Architecture

- The Use Case View is putting it all together. The elements in the four views are shown to work together by using a set of scenarios. The scenarios are abstractions of the most important functionalities (requirements). Scenarios are expressed by using Objects and Interaction diagrams.
General formula of Interfaces

The total number of interfaces: \( 2^n - n - 1 \)

The number of second order interfaces: \( C_2^n = \frac{n!}{(n-2)! \cdot 2!} \)

The number of third order interfaces: \( C_3^n = \frac{n!}{n(-3)! \cdot 3!} \)

The total number of possible outcomes: \( 2^n = \sum_{m=0}^{n} C_m^n \)

** A System that contains all intersections is called Logically Integrated **