The Method-Framework for Engineering System Architectures (MFESA)

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Donald Firesmith
System and Software Technology Conference (SSTC)
Salt Lake City, Utah
20-23 April 2009
**Tutorial Objectives**

Introduce attendees to the Method Framework for Engineering System Architectures (MFESA):

- MFESA *Ontology* of reusable concepts and terminology
- MFESA *Metamodel* of reusable method components
- MFESA *Repository* of reusable method components:
  - MFESA Architectural Work Units and Work Products
  - MFESA Architectural Workers
- MFESA *Metamethod* for generating appropriate project-specific system architecture engineering methods

Thereby improve the attendees’ system architecture engineering methods and associated processes (process improvement)
MFESA Project

Started January 2007

Collaborators:

- SEI Acquisition Support Program (ASP) – Don Firesmith (Team Lead), Peter Capell, Bud Hammons, and Tom Merendino
- MITRE – Dietrich Falkenthal (Bedford MA)
- USAF – DeWitt Latimer (USC)

Current work products:

- Tutorials and Training Materials
- Articles

Eventual work products (we hope!):

- Informational website with method components and associated tools
Intended Tutorial Attendees

System and Subsystem Architects
Process Engineers
Requirements Engineers
Technical and Administrative Managers
Acquirers
Developers
Testers
Trainers and Educators
Standards Developers
Academic Researchers
Any other Stakeholders
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

MFESA Repository of Reusable Method Components
  • Architectural Work Units and Work Products
  • Architectural Workers

MFESA Metamethod

Conclusion
System Architecture – Traditional Definition

System Architecture

the organization of a system including its major components, the relationships between them, how they collaborate to meet system requirements, and principles guiding their design and evolution

Note that this definition is primarily oriented about the system’s structure.

Yet systems have many static and dynamic logical and physical structures.
System Architecture – MFESA Definition

System Architecture

all of the most important, pervasive, top-level, strategic decisions, inventions, engineering tradeoffs, assumptions, and their associated rationales concerning how the system will meet its derived and allocated requirements

Includes:

• All major logical and physical and static and dynamic structures
• Other architectural decisions, inventions, tradeoffs, assumptions, and rationales:
  — Approach to meet quality requirements
  — Approach to meet data and interface requirements
  — Architectural styles, patterns, mechanisms
  — Approach to reuse (build/buy decisions)
• Strategic and pervasive design-level decisions
• Strategic and pervasive implementation-level decisions
## Architecture vs. Design

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Design</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Pervasive</em> (Multiple Components)</td>
<td><em>Local</em> (Single Components)</td>
</tr>
<tr>
<td><em>Strategic</em> Decisions and Inventions</td>
<td><em>Tactical</em> Decisions and Inventions</td>
</tr>
<tr>
<td><em>Higher-Levels</em> of System</td>
<td><em>Lower-Levels</em> of System</td>
</tr>
<tr>
<td><em>Huge Impact</em> on Quality, Cost, &amp; Schedule</td>
<td><em>Small Impact</em> on Quality, Cost, &amp; Schedule</td>
</tr>
<tr>
<td><em>Drives</em> Design and Integration Testing</td>
<td><em>Drives</em> Implementation and Unit Testing</td>
</tr>
<tr>
<td><em>Driven by</em> Requirements and Higher-Level</td>
<td><em>Driven by</em> Requirements, Architecture, and</td>
</tr>
<tr>
<td><em>Architecture</em></td>
<td><em>Higher-Level Design</em></td>
</tr>
<tr>
<td><em>Mirrors</em> Top-Level Development Team Organization</td>
<td><em>No Impact</em> on</td>
</tr>
<tr>
<td><em>(Conway’s Law)</em></td>
<td><em>Top-Level Development Team Organization</em></td>
</tr>
</tbody>
</table>
System Architecture Engineering

System Architecture Engineering

the subdiscipline of systems engineering consisting of all architectural work units performed by architectural workers (architects, architecture teams, and their tools) to develop and maintain architectural work products (including system or subsystem architectures and their representations)
System Architecture is Critical

Supports achievement of critical architecturally significant requirements

Greatly affects cost and schedule

Enables engineering of system quality characteristics and attributes

Drives all downstream activities
System Architecture Engineering is critical to Project Success

Limitations of Current Methods and Standards

Do not adequately address:

- The increasing size and complexity of many current systems
- All types of architectural components (e.g., software)
- All types of interfaces (interoperability and intraoperability)
- All potentially important system structures, views, models, and other architectural representations
- All life cycle phases (production, evolution, and maintenance of architectural integrity)
- System quality characteristics, attributes, and requirements
- Reuse and Component-Based Development (CBD)
- Specialty engineering areas (such as safety and security)
More Limitations of Current Methods and Standards

Current methods:

- Overemphasize two structures:
  - Static logical functional decomposition view
  - Static physical aggregation decomposition view
- Are weak on structure, view, and model consistency.
- Confuse requirements engineering with architecture engineering.
- Tend to assume that One Size Fits All.
- Produce only a single architectural vision.
- Excessively emphasize architectural models over other architectural representations.
Architecture Engineering Challenges

How good is ‘Good enough’?

We lack sufficient adequately trained and experienced architects.

- Many young architects must perform tasks for which many are under qualified.

Architects may use multiple inconsistent architecture engineering methods.

Architecture engineering methods are often incomplete or incompletely documented.

Architects can rely too much on architectural engineering tools.
Architecture Engineering Challenges

Different stakeholders have different and possibly conflicting needs for different architectural representations at different levels of abstractions:

- **Requirements Engineers** – Ensure architecturally significant (e.g., quality) requirements are properly engineered
- **Architects** – Capture and convey their architecture to themselves, other architects, and other stakeholders
- **Designer and Implementers** – Constrain designs and implementations
- **Specialty Engineers** – ensure architecture supports specialty engineering requirements and incorporates related patterns/mechanisms.
- **Testers** – Integration tests and whitebox system and component testing
- **Manufacturers** – Producibility of the system given its architecture
- **Acquirers and Funders** – Understand what is being acquired and paid for
- **Managers** – Manage development and Conway’s Law
- **Certifiers, Accreditors, and Regulators** – Ensure system will be able to be safely and securely operated

Size (small through ultra-large-scale)
Complexity
Autonomy of subsystems (useful, self-contained, not controlled by others)
Criticality (business, safety, and security of system and individual subsystems)
Domains (such as aviation, telecommunications, weapons)
Driven by requirements (top-down) or subsystem availability (bottom-up)
Emergent behavior and characteristics (necessary, beneficial, foreseeable)
Geographical distribution of subsystems

Homogeneity/heterogeneity of subsystems
Intelligence
Operational dependence on other systems
Reconfigurability (adding, replacing, or removing subsystems)
Relative amounts of hardware, software, people, facilities, manual procedures, ...
Requirements (existence, volatility, quality characteristics and attributes, constraints)
Self-regulation (proactive vs. reactive, homeostasis)
Synergism/independence of subsystems
Technologies used (including diversity, maturity, and volatility)
Why Method Engineering? – Organizations Vary Greatly

Number of organizations
Size of organizations
Types of organizations:
  • Owner, Acquirer, Developer, Operator, User, Maintainer
  • Prime contractor, subcontractors, vendors, system integrator
Degree of centralized/distributed governance:
  • Authority, policy, funding, scheduling
  • Directed, Acknowledged, Collaborative, or Virtual
Management and engineering culture
Geographical distribution
Staff expertise and experience
Why Method Engineering? – Endeavors Vary Greatly

Type (project, program of projects, enterprise)

Contracting:
  • Formality
  • Type (e.g., fixed-price or cost plus fixed fee)

Lifecycle scope (development, manufacturing, sustainment)

System scope (subsystem, system, “system of systems”)

Duration (weeks, months, years, or decades)

Schedule (adequacy, criticality, coordination)

Funding (adequacy, distribution)
Why Method Engineering? – Stakeholders Vary Greatly

Type of stakeholders:

- Acquirer, developer, maintainer, member of the public, operator, regulator, safety/security accreditor/certifier, subject matter expert, user, …

Number of stakeholders

Authority (requirements, funding, policy, …)

Accessibility of the stakeholders to the architecture teams

Volatility of stakeholder turnover (especially acquirers)

Motivation and needs
Why Method Engineering? – Bottom Line

No *single* system architecture engineering method is sufficiently general and tailorable to meet the needs of all endeavors.

Method engineering enables the creation of appropriate, system/organization/endeavor/stakeholder-specific architecture engineering methods.
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

MFESA Repository of Reusable Method Components

• Architectural Work Units and Work Products
• Architectural Workers

MFESA Metamethod

Conclusion
Definition

Method-Framework for Engineering System Architectures (MFESA)

a method framework for engineering appropriate situation-specific system architecture engineering (SAE) methods

MFESA is not a single system architecture engineering method.
As-Performed Processes
As-Intended Methods
Method Frameworks
Primary Inputs to MFESA

- ANSI/EIA 632-2003
- ANSI/IEEE 1471-2000
- Department of Defense Architecture Framework (DODAF)
- INCOSE SE Handbook
- ISO/IEC 15288-2002
- Naval Systems Engineering Guide
- SEI Attribute-Driven Design (ADD)
- SEI Capability Maturity Model Integrated (CMMI)
- SEI Evolutionary Process for Integrating COTS-based Systems (EPIC)
- SEI System Architecture Engineering Experience

MFESA
MFESA Components (Top View)

MFESA

Method Engineering Framework

MFESA Ontology

MFESA Metamodel

MFESA Repository

MFESA Metamethod
MFESA Components (Detailed View)
MFESA Addresses Size and Complexity

- First Generation
  - General Purpose Individual Standards and Methods
- Second Generation
  - Method Frameworks and Project-Specific Methods
- Third Generation
  - Approaches Needed

Maximum Size and Complexity of the System and its Architecture

Date in Years

Today
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

MFESA Repository of Reusable Method Components
  • Architectural Work Units and Work Products
  • Architectural Workers

MFESA Metamethod

Conclusion
MFESA Ontology

More than merely an architectural glossary
Information model of system architecture engineering
Defines foundational architectural concepts and terminology
Defines relationships between concepts
MFESA Ontology of Concepts and Terminology

System
System Architecture
Architectural Structures
Architectural Styles, Patterns, and Mechanisms
Architectural Drivers and Concerns
Quality Model, Quality Requirements,
Architectural Representations
Architectural Models, Structures, Views, and Focus Areas
Architectural Quality Cases
Architectural Visions
System - Definition

System

a cohesive integrated set of system components (i.e., an aggregation structure) that collaborate to provide the behavior and characteristics needed to meet valid stakeholder needs and desires

Important Ideas:

- Modeled as hierarchical aggregate structure
- Integrated system components
- Components collaborate
- Emergent behavior and properties
System Component Types

Subsystems
Consumable materials (e.g., ammunition, fuel, lubricants, reagents, and solvents)
Data
Documentation (both separate physical and built-in electronic documentation)
Equipment (e.g., maintenance, support, and training equipment)
Facilities (e.g., maintenance, manufacturing, operations, support, training, and disposal facilities including their component property, buildings, and their furnishings)
Hardware
Manual procedures
Networks (for the flow of data, power, and material)
Organizations
Personnel
Physical interfaces
Software
Tools
System – Partial Example

Aircraft System of Systems

- Airframe Segment
  - Empennage
    - Horizontal Stabilizers
    - Vertical Stabilizer
    - Tail Cone
  - Fuselage
    - Doors
    - Windows
    - Skin
    - Structure
  - Wings

- Avionics Segment
  - Auto Flight
    - Communications
    - Crew Interface
    - Entertainment
    - Information Processing
  - Navigation
  - Prognostics and Health Management
  - Sensors

- Interiors Segment
  - Crew Compartment
    - Passenger Compartments
    - Cargo Compartments
    - Galley
  - Lavatories
  - Emergency Provisions
  - Water & Waste
  - Environment

- Propulsion Segment
  - Engines
    - Fuel
    - Nacelles
    - Pylons
  - Air Conditioning
  - Air Pressure
  - Oxygen

- Vehicle Segment
  - Auxiliary Power
    - Electrical Power
    - Fire Protection
    - Flight Control Surfaces
      - Ailerons
      - Elevators
      - Flaps
      - Rudder
    - Hydraulic Power
    - Pneumatic Power
    - Landing Gears
    - Shipside Lighting
Some System Characteristics

Multiple Components
Multiple Interactions between Components
Multiple Structures (Logical and Physical, Static and Dynamic)
Multiple:
  • Views and Viewpoints
  • Models
  • Focus Areas
What about Systems of Systems?

System of Systems (SOS)

a system composed of systems

Almost all systems are composed of systems (i.e., subsystems)
When most people say *systems of systems*, what they really mean is something like this:

an ultra-large and complex, highly flexible, dynamically evolving, technologically ambitious, and geographically-distributed *system of pre-existing, heterogeneous, autonomous, self-contained, and independently governed (e.g., acquired, developed, operated, scheduled, and funded)* systems, whereby the system of systems exhibits significant amounts of unexpected emergent behavior and characteristics

Engineering the architecture of such systems of systems calls for a different architecture engineering method than simpler systems.
System and System Architecture - Ontology

- Subsystem
- System of Systems
- System
  - abstracts the
  - System Architecture
- Architect(s)
  - engineer the
- Architectural Decisions
- Architectural Inventions
- Architectural Tradeoffs
- Architectural Assumptions
- Associated Rationales
Architectural Structure, Element, and Component – Definitions

Architectural Structure

a cohesive set of architectural elements connected by associated relationships that captures a set of related architectural decisions, inventions, tradeoffs, assumptions, and rationales

Architectural Element

a part of an architectural structure

Architectural Component

a physical architectural element of a static physical aggregation structure
Architectural Structure - Ontology

- System
  - 1
  - 1
  - consists primarily of
  - drive

- System Architecture
  - abstracts the
  - 1
  - 1

- Architectural Structures
  - 1..*
  - 1..*
  - drive and constrain
  - may have known
  - 0..*
  - incorporated most

- Architectural Elements
  - 1..*

- Relationships Between Architectural Elements
  - 1..*

- Static Structures
- Dynamic Structures
- Logical Structures
- Physical Structures

- Architectural Decisions
- Architectural Inventions
- Architectural Tradeoffs
- Architectural Assumptions
- Associated Rationales
Architectural Styles, Patterns, and Mechanisms - Definitions

Architectural Pattern

a well-documented reusable solution to a commonly occurring architectural problem within the context of a given set of existing architectural concerns, decisions, inventions, engineering trade-offs, and assumptions

Architectural Style

a top-level architectural pattern that provides an overall context in which lower-level architectural patterns exist

Architectural Mechanism

a major architectural decision or invention, often an element of an architectural pattern
Architectural Styles, Patterns, and Mechanisms - Ontology

Architectural Styles <<use of>>

Architectural Patterns <<use of>>

Architectural Mechanisms <<use of>>

Architectural Decisions

incorporate most

1..*

consists primarily of the

1..*

are abstractions of the

1..*

System Architecture

1

System

1

1
Architectural Drivers and Concerns - Definitions

Architectural Driver

an architecturally significant product or process requirement that drives the engineering of the system architecture

Architectural Concern

a *cohesive* collection of architectural drivers
Architectural Drivers and Concerns - Ontology

Architectural Focus Areas

Architectural Concerns

Architectural Drivers

Architecturally Significant Product Requirements

Architecturally Significant Process Requirements

Architectural Structures

Static Structures

Dynamic Structures

Logical Structures

Physical Structures

System

drive and constrain
are abstractions of the
1..^n

Architectural Elements

1..^n

correct

Relationships Between Architectural Elements

drive the engineering of the

System Architecture

1

MFESA Tutorial
Donald Firesmith, 22 April 2009
© 2009 Carnegie Mellon University
Architectural Concern – An Example
MFESA Quality Model

Architectural Components

System

defines the meaning of the quality of a

defines the meaning of a specific type of quality of a

Quality Characteristics

Quality Attributes

Quality Measurement Scales

Quality Measurement Method

are measured along

measures quality along

are measured using

Internal Quality Characteristics

External Quality Characteristics
Internal Quality Characteristics
External Quality Characteristics

Quality Characteristic

Internal Quality Characteristic

Configurability
Compliance
Robustness
Defensibility
Safety

Efficiency
Dependability
Performance
Tolerance
Security

Functionality
Environmental Compatibility
Soundness
Availability
Survivability

Operability
Interoperability
Correctness
Capacity

Usability
Serviceability
Reliability
Predictability
Stability
Example Characteristic and Attributes
Example Characteristic and Attributes
Quality Requirements

states stakeholders importance of achieving a

Quality Goal

quantifies a

defines stakeholders minimum acceptable level of quality of a

Subsystem

Quality Requirement

Quality Goal

Condition

is applicable during

determines existence of

Quality Characteristic

Quality Attribute

Quality Criterion

shall exceed

Quality Measure

Quality Threshold

is measured along a

is measured using a

defines the meaning of the quality of a

Quality Metric

Quality Model

System
Architectural Representations - Definition

Architectural Representation

a cohesive collection of information that documents a system architecture

Not the same thing as the architecture
Architectural Representations - Ontology
Architectural Models, Views, and Focus Areas - Definitions

Architectural Model

an architectural representation that abstracts a single system structure in terms of the structure’s architectural elements and the relationships between them

Architectural View

an architectural representation describing a single architectural structure of a system consisting of one or more related models of that structure

Architectural Focus Area

an architectural representation consisting of the cohesive set of all architectural decisions, decisions, and tradeoffs related to a specific architectural concern, regardless of the architectural view, model, or structure where they are documented or found
Architectural Models, Views, and Focus Areas - Ontology

- Architectural Representations
- Architectural Descriptions
- Architectural Views
- Architectural Viewpoint
- Quality Focus Areas
- Architectural Models
- Architectural Structures
- System Architecture
- Quality Attributes
- Quality Characteristics

Arrows indicate relationships and cardinalities. Each box represents a concept with relationships to other concepts.
Multifaceted architecture having multiple structures requiring multiple models providing multiple views.

Architects must ensure view and model consistency.

Data Flow View

Mode and State View

Logical Functional Decomposition View

Physical Decomposition View

Collaboration View

Information View

Services View
Quality Cases

Work Product

make developer’s’ case for adequate quality of the

justify belief in

Quality Case

supports

Evidence

Arguments

Claims

is developed for

Quality Characteristic

Quality Attribute
Architectural Quality Cases

System/Subsystem Architecture

makes architects’ case for adequate quality of the

Architectural Claims:
Architecture Helps System Meet its Quality Requirements

Architectural Arguments:
Architecture includes Architectural Decisions, Inventions, Tradeoffs, Assumptions, and Rationales

Architectural Evidence:
Official Architectural Representations (e.g., Architectural Diagrams, Models, Documents) and Witnessed Demonstrations

is developed for

Quality Characteristic

Quality Attribute supports

Architectural Quality Case

justify belief in
Architectural Quality Case Diagram
Example Architectural Quality Case Diagram
Architecture Visions and Vision Components - Definitions

Architectural Vision

one of the more important actual or potential architectural decisions, inventions, or tradeoffs addressing one or more architectural concerns

Architectural Vision Component

one of the more important actual or potential architectural decisions, inventions, or tradeoffs addressing one or more architectural concerns

Note that multiple candidate architectural visions are often created before one is selected and completed to produce the actual architecture
Architecture Visions and Vision Components - Ontology

- Architectural Representations
- Architectural Descriptions
- Architectural Visions
  - Architectural Vision Components
- Document architects' initial visions of the
- Document some of the most important parts of the candidate

- System Architecture
  - Architectural Decisions
  - Architectural Inventions
  - Architectural Tradeoffs
  - Architectural Assumptions
  - Associated Rationales

Drive
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

**MFESA Metamodel of Reusable Method Components**

MFESA Repository of Reusable Method Components

- Architectural Work Units and Work Products
- Architectural Workers

MFESA Metamethod

Conclusion
A Metamodel is a Model of a Model.

MFESA Metamodel defines three Foundational Types of Reusable Method Components.

Based on OPEN Process Framework Metamodel.

Simplification of ISO/IEC 24744

Not based on OMG Metamodel.
System Architecture Engineering – Methods and Processes

System Architecture Engineering Method

a systematic, documented, intended way that system architecture engineering should be performed

System Architecture Engineering Process

an actual way that system architecture engineering is performed in practice on an endeavor
Method Engineering Models

- **Process Metamodel**
  - **As-Intended Method (Process Model)**
    - **As-Performed Process**
  - **Metamethod Components**
    - **Method Components**
      - **Instantiation (instance of)**
      - **specialization (inheritance)**

models

specifies
Method vs. Process

System Architecture Engineering
- documents intended way to perform
- Is the actual performance of

System Architecture Engineering Method
- documents the intended

System Architecture Engineering Process
- consists of instances of

Method Components
- System Architecture Engineering

Architectural Workers
- perform
- produce
- create and modify

Architectural Work Units

Architectural Work Products

MFESA Tutorial
Donald Firesmith, 22 April 2009
© 2009 Carnegie Mellon University
MFESA Metamodel of Reusable Method Components

**MFESA Repository**
- stores the

**MFESA Reusable Method Components**
- perform
  - create and update
  - produce

**Architecture Teams**
- membership
  - Architects
  - Architecture Tools

**Architecture Work Units**
- use
  - **Architectural Work Products**
  - **Architecture Process Work Products**

**Architecture Engineering Discipline**
- Architecture Engineering Tasks
- Architecture Engineering Techniques

**Architecture Workers**
- use

**Architecture Representations**
- describe
  - Architectures
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

**MFESA Repository of Reusable Method Components**

- Architectural Work Units and Work Products
- Architectural Workers

MFESA Metamethod

Conclusion
MFESA Repository

Stores reusable system architecture engineering method components:

- Architecture Work Units
- Architecture Work Products
- Architecture Workers

Should provide easy access to method components:

- Identification and selection of relevant method components
- Tailoring of selected method components
- Configuration management of method components
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

MFESA Repository of Reusable Method Components
  • Architectural Work Units and Work Products
  • Architectural Workers

MFESA Metamethod

Conclusion
MFESA Tasks

T1: Plan and Resource the Architecture Engineering Effort

T2: Identify the Architectural Drivers

T3: Create the First Versions of the Most Important Architectural Models

T4: Identify Opportunities for the Reuse of Architectural Elements

T5: Create the Candidate Architectural Visions

T6: Analyze Reusable Components and their Sources

T7: Select or Create the Most Suitable Architectural Vision

T8: Complete and Maintain the Architecture

T9: Evaluate and Accept the Architecture

T10: Ensure Architectural Integrity
## Effort by MFESA Task

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Phase  (time)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Initiation</td>
</tr>
<tr>
<td>1 Plan and Resource the Architecture Engineering Effort</td>
<td></td>
</tr>
<tr>
<td>2 Identify the Architectural Drivers</td>
<td></td>
</tr>
<tr>
<td>3 Create First Versions of the Most Important Architectural Models</td>
<td></td>
</tr>
<tr>
<td>4 Identify Opportunities for the Reuse of Architectural Elements</td>
<td></td>
</tr>
<tr>
<td>5 Create the Candidate Architectural Visions</td>
<td></td>
</tr>
<tr>
<td>6 Analyze the Reusable Components and their Sources</td>
<td></td>
</tr>
<tr>
<td>7 Select or Create the Most Suitable Architectural Vision</td>
<td></td>
</tr>
<tr>
<td>8 Complete and Maintain the Architecture</td>
<td></td>
</tr>
<tr>
<td>9 Evaluate and Accept the Architecture</td>
<td></td>
</tr>
<tr>
<td>10 Ensure Architectural Integrity</td>
<td></td>
</tr>
</tbody>
</table>
Plan, Prepare, Act, and Check

**PLAN**
- T1: Plan and Resource the Architecture Engineering Effort

**PREPARE**
- T2: Identify the Architectural Drivers
- T3: Create the First Versions of most Important Architectural Models
- T4: Identify Opportunities for the Reuse of Architectural Elements

**CHECK**
- T9: Evaluate and Approve the Architecture
- T10: Ensure Architectural Integrity

**ACT**
- T5: Create Candidate Architectural Visions
- T6: Analyze Reusable Components and their Sources
- T7: Select or Create the Most Suitable Architectural Vision
- T8: Complete and Maintain the Architecture
Concurrent MFESA Tasks
Architectural Visions - Flow

Task: T5: Create Initial Architecture Models

Architecture Model 1 WP  Architecture Model 2 WP  ...  Architecture Model N WP

Task: T6: Create Candidate Architectural Visions

Candidate Vision 1 WP  Candidate Vision 2 WP  ...  Candidate Vision N WP

Task: T7: Select or Create Most Suitable Architectural Vision

Selected Vision 1a WP  Selected Vision 1b WP  ...  Selected Vision 1n WP

Task: T8: Complete and Iterate the Architectures
MFESA Tasks Supporting Reuse

T4: Identify Opportunities for the Reuse of Architectural Elements
- Identify and analyze existing architectural representations for potential reuse:
  - Legacy, variant, and competing systems
  - Reference and enterprise architectures
- Identify architectural patterns and styles
- Identify candidate reusable architectural elements
- Initiate early relationships with potential suppliers

T6: Analyze Reusable Components and Their Sources
- Identify, characterize, evaluate, and conditionally select reusable components and their sources

T8: Complete and Maintain the Architecture (and Its Representations)
- Select, acquire, baseline, and integrate the reusable architectural components
MFESA Task 1) Plan and Resource the Architecture Engineering Effort

Task 1) Plan and Resource Architecture Engineering Effort

Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models
Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources
Task 7) Select or Create Most Suitable Architectural Vision
Task 8) Complete and Maintain the Architecture
Task 9) Evaluate and Accept the Architecture
Task 10) Ensure Architectural Integrity
MFESA Task 1) Plan and Resource the Architecture Engineering Effort

Goal:

• Prepare the system engineering team to engineer the system architecture and its representations.

Objectives:

• Staff and train system architecture teams to engineer the system architecture.
• Develop and document the system architecture engineering method.
• Develop plans, standards, and procedures for engineering the system architecture.
• Prioritize and schedule the system architecture engineering effort.
MFESA Task 1) Plan and Resource the Architecture Engineering Effort

**Inputs:**
- Request for proposal
- Project contract
- Project charter
- System vision statement
- System concept of operations (ConOps)
- System operational requirements document
- System requirements specification
- Reference architecture
- Enterprise architecture
- MFESA references

**Steps:**
1. Staff the system architecture team(s).
2. Select or instantiate and tailor one or more MFESA-compliant methods.
3. Select architecture modeling methods.
4. Evaluate and select the architecture engineering tools.
5. Provide training in architecture engineering.
6. Develop the system architecture plan.
7. Develop the architecture engineering conventions.
8. Prioritize and schedule the system architecture engineering effort.

**Outputs:**
- Architecture team charters
- Architecture engineering conventions
- Architecture engineering tool evaluation team charter
- Architecture engineering tool evaluation report(s)
- Architecture engineering training materials
- Architecture plan(s)
- Architecture engineering schedule
- Architectural risks and opportunities
MFESA Task 1) Plan and Resource the Architecture Engineering Effort

Guidelines

- Properly staff the top-level architecture team(s).
- Properly plan the architecture engineering effort.
- Produce and maintain a proper and sufficient schedule.
- Reuse or create appropriate MFESA method(s).
- Select appropriate architecture modeling method(s).
- Select appropriate architecture engineering tools.
- Provide appropriate training.
MFESA Task 1) Plan and Resource the Architecture Engineering Effort

Pitfalls

• Architects produce incomplete architecture plans and conventions.
• Management provides inadequate resources.
• Management provides inadequate staff and stakeholder training.
• Architects lack authority.
• Architects instantiate the entire MFESA repository without tailoring.
• Tool vendors drive architecture engineering and modeling methods.
• Planning and resourcing are unsynchronized.
• Planning and resourcing are only done once up front.
MFESA Task 2)
Identify the Architectural Drivers

Task 1) Plan and Resource Architecture Engineering Effort

Task 2) Identify the Architectural Drivers

Task 3) Create Initial Architectural Models

Task 4) Identify Opportunities for Reuse of Architectural Elements

Task 5) Create Candidate Architectural Visions

Task 6) Analyze Reusable Components and their Sources

Task 7) Select or Create Most Suitable Architectural Vision

Task 8) Complete and Maintain the Architecture

Task 9) Evaluate and Accept the Architecture

Task 10) Ensure Architectural Integrity
MFESA Task 2) Identify the Architectural Drivers

Goal:

• Identify the architecturally significant product and process requirements that drive the development of the system architecture.

Objectives:

• Understand and verify the product and process requirements that have been allocated to the system or subsystem being architected.
• Categorize sets of related architecturally significant requirements into cohesive architectural concerns.
• Provide a set of architectural concerns to drive the:
  — Identification of potential opportunities for architectural reuse.
  — Analysis of potentially reusable components and their sources.
  — Creation of an initial set of draft architectural models.
  — Creation of a set of competing candidate architectural visions.
  — Selection of a single architectural vision judged most suitable.
  — Completion and maintenance of the resulting system architecture.
  — Evaluation and acceptance of the system architecture.
MFESA Task 2) Identify the Architectural Drivers

**Inputs:**
- Request for proposal
- System vision statement
- System concept of operations (ConOps)
- System operational requirements document
- System requirements repository including relevant product and process requirements
- System requirements specification
- System requirements models (e.g., use case)
- System requirements evaluation results
- Security policy
- Known architectural risks and opportunities

**Steps:**
1. Collaborate to help engineer the architecturally significant requirements.
2. Identify and label the architecturally significant requirements.
3. Verify the potentially relevant requirements.
4. Collaborate to fix requirements defects.
5. Identify the architectural concerns.
6. Evaluate and iterate the architectural concerns.
7. Identify any new architectural risks and opportunities.

**Outputs:**
- Requirements recommendations
- Architectural concerns
- Requirements metadata
- New and updated architectural risks and opportunities
MFESA Task 2)
Identify the Architectural Drivers

Guidelines

• Collaborate closely with the requirements team.
• Notify the requirements team(s) of relevant requirements defects.
• Consider the impact of the architecture on the requirements.
• Respect team boundaries and responsibilities.
• If necessary, clarify relevant requirements with the stakeholders.
• Concentrate on the architecturally significant requirements.
• Quality attributes can be architectural concerns too.
• Formally manage architectural risks.
MFESA Task 2) Identify the Architectural Drivers

Pitfalls

- All requirements are architecturally significant.
- Well-engineered architecturally significant requirements are lacking.
- Architects rely excessively on functional requirements.
- The architects ignore the architecturally significant functional and process requirements.
- Specialty engineering requirements are misplaced and ignored.
- Unnecessary constraints are imposed on the architecture.
- Architects engineer architecturally significant requirements.
- Requirements lack relevant metadata.
- Architects fail to clarify architectural drivers.
MFESA Task 3) Create Initial Architectural Models

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers

**Task 3) Create Initial Architectural Models**

Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources
Task 7) Select or Create Most Suitable Architectural Vision
Task 8) Complete and Maintain the Architecture
Task 9) Evaluate and Accept the Architecture
Task 10) Ensure Architectural Integrity
MFESA Task 3) Create Initial Architectural Models

Goal:

• Create an initial set of partial draft architectural models of the system architecture.

Objectives:

• Capture the most important candidate elements of the eventual system architecture (i.e., architectural decisions, inventions, trade-offs, assumptions, and rationales).

• Provide the most important views and focus areas of the system architecture.

• Ensure that these candidate architectural elements sufficiently support the relevant architectural concerns.

• Provide a foundation of architectural models from which to create a set of competing candidate architectural visions.
MFESA Task 3)
Create Initial Architectural Models

Inputs:
- Architectural concerns (sets of architecturally-significant requirements)
- Requirements metadata
- Candidate reusable architectural elements
- Candidate architectural visions and vision components
- Known architectural risks and opportunities

Steps:
1. Identify the relevant architectural structures.
2. Select the appropriate architectural viewpoints, views, and models.
3. Select the appropriate focus areas.
4. Collaborate with specialty engineering groups and other stakeholders.
5. Develop initial partial competing models of the architectural structures.
6. Conditionally allocate the architectural concerns to the underlying component types.
7. Identify the associated potentially relevant technologies.
8. Perform architectural tradeoff analyses.
9. Evaluate the architectural models and associated documentation.
10. Identify any new architectural risks and opportunities.

Outputs:
- Initial draft architecture
- Initial partial draft architectural models capturing relevant architectural views
- Initial partial draft representations (e.g., architectural quality cases) for relevant focus areas
- New and updated architectural risks and opportunities
MFESA Task 3) Create Initial Architectural Models

Guidelines

• Perform architectural trade-off analysis.
• Reuse architectural principles, heuristics, styles, patterns, vision components, and metaphors.
• Use iterative, incremental, and parallel development.
• Begin developing logical models before physical models and static models before dynamic models.
• Do not overemphasize the physical decomposition hierarchy.
• Use explicitly documented system partitioning criteria.
• Model concurrency.
• Consider the impact of hardware decisions on usability and software.
• Consider human limitations when allocating system functionality to manual procedures.
• Do not start from scratch.
• Formally manage architectural risks.
MFESA Task 3)
Create Initial Architectural Models

Pitfalls

- The architects succumb to analysis paralysis.
- The architects engineer too few architectural models.
- The architects engineer inappropriate models and views.
- The architects construct views but no focus areas.
- Some stakeholders believe that the models are the architecture.
- Inconsistencies exist between models, views, and focus areas.
- The architects use inappropriate architectural patterns.
- System decomposition is performed by the acquisition organization.
MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models

**Task 4) Identify Opportunities for Reuse of Architectural Elements**

Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources
Task 7) Select or Create Most Suitable Architectural Vision
Task 8) Complete and Maintain the Architecture
Task 9) Evaluate and Accept the Architecture
Task 10) Ensure Architectural Integrity
MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements

Goal:

- Identify any opportunities to reuse existing architectural work products as part of the architecture of the target system or subsystem being developed. Any opportunities so identified become a collection of reusable architectural element candidates.

Objectives:

- Identify the architectural risks and opportunities for improving the architectures associated with the relevant legacy or existing system(s) should they be selected for reuse and incorporation within the target environment.
- Identify any additional architectural concerns due to the constraints associated with having legacy or existing architectures.
- Understand the relevant legacy or existing architectures sufficiently well to identify potentially reusable architectural elements.
- Provide a set of reusable architectural element candidates to influence (and possibly include in) a set of initial draft architectural models.
MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements
MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements

**Inputs:**
- Architectural concerns
- Draft versions of the most important architectural models
- Candidate architectural visions and vision components
- Other existing representations of the current architecture
- Representations of pre-existing architectures
- Architectural patterns
- Known architectural risks and opportunities

**Steps:**
1. Identify architectural concerns that may be implemented via reuse.
2. Identify and analyze the architectural representations of the prior version of the system or subsystem.
3. Identify and analyze the architectural representations of existing variants of system or subsystem.
4. Identify and analyze the architectural representations of any competing systems or subsystems.
5. Identify and analyze the system's product line reference architecture.
6. Identify and analyze the organization's enterprise reference architecture(s).
7. Identify and analyze any industry standard architecture(s).
8. Identify potentially reusable architectural patterns.
9. Identify candidate potentially-reusable architectural elements.
10. Initiate early relationships with potential suppliers of reusable components.
11. Update the architectural concerns.
12. Identify any new architectural risks and opportunities.

**Outputs:**
- Candidate reusable architectural elements
- Updated architectural concerns
- New and updated architectural risks and opportunities
MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements

Guidelines

- Do not start from scratch.
- Do not be excessively constrained by the past.
- Conform to the enterprise architecture.
- Conform to the product line reference architecture.
- Consider system architecture patterns.
- Identify opportunities for reuse in the architectural models.
- Formally manage architectural risks.
MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements

Pitfalls

- The architects start from scratch.
- The architects ignore past lessons learned.
- The architects over-rely on previous architectures.
- The architects select specific OTS components too early.
- The architects assume reuse of immature architectural components.
- The architects assume the reuse of immature technologies.
- Inadequate information exists to determine reusability.
MFESA Task 5) Create Candidate Architectural Visions

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models
Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources
Task 7) Select or Create Most Suitable Architectural Vision
Task 8) Complete and Maintain the Architecture
Task 9) Evaluate and Accept the Architecture
Task 10) Ensure Architectural Integrity
MFESA Task 5)
Create Candidate Architectural Visions

Goal:

- Create multiple candidate architectural visions of the system architecture.

Objectives:

- Verify that the candidate subsystem architectural visions sufficiently support the relevant architecture concerns.
- Provide a sufficiently large and appropriate set of competing candidate architectural visions from which a single vision may be selected as most suitable.
MFESA Task 5)
Create Candidate Architectural Visions

Inputs:
- Architectural concerns
- Initial draft versions of the architectural models
- Candidate reusable architectural elements
- Known architectural risks and opportunities

Steps:
1. Identify potentially usable architectural vision components.
2. Create and document the competing architectural visions.
3. Identify vision pros and cons.
4. Verify the architectural visions.
5. Iterate the architectural visions.
6. Identify any new architectural risks and opportunities.

Outputs:
- Candidate architectural vision components
- Architectural vision component versus vision matrix
- Architectural concern versus vision component matrix
- Competing architectural visions list
- Draft architectural vision documents
- New and updated architectural risks and opportunities
# MFESA Task 5)
## Create Candidate Architectural Visions

<table>
<thead>
<tr>
<th>Architectural Vision Component vs. Architectural Vision Matrix</th>
<th>Candidate Architectural Visions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Architectural Vision 1</td>
</tr>
<tr>
<td>Component 1</td>
<td>X</td>
</tr>
<tr>
<td>Component 2</td>
<td>X</td>
</tr>
<tr>
<td>Component 3</td>
<td>X</td>
</tr>
<tr>
<td>Component 4</td>
<td>X</td>
</tr>
<tr>
<td>Component 5</td>
<td>X</td>
</tr>
<tr>
<td>Component 6</td>
<td>X</td>
</tr>
<tr>
<td>Component 7</td>
<td>X</td>
</tr>
<tr>
<td>Component 8</td>
<td>X</td>
</tr>
<tr>
<td>Component 9</td>
<td>X</td>
</tr>
<tr>
<td>Component 10</td>
<td>X</td>
</tr>
<tr>
<td>Component 11</td>
<td>X</td>
</tr>
<tr>
<td>Component 12</td>
<td>X</td>
</tr>
<tr>
<td>Component 13</td>
<td>X</td>
</tr>
</tbody>
</table>
**MFESA Task 5**
Create Candidate Architectural Visions

Example Architectural Concern vs. Vision Component Matrix

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Control</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Integrity (Message)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>Integrity (Software)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>Integrity (Data)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>Nonrepudiation</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>0</td>
</tr>
<tr>
<td>Availability</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Cost</td>
<td>++</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Performance</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Usability</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>++</td>
<td></td>
</tr>
</tbody>
</table>
MFESA Task 5)
Create Candidate Architectural Visions

Guidelines

• Complete candidate architectural visions to appropriate level of detail.

• Prepare architectural components for OTS incorporation.

• Identify an appropriate number of candidate architectural visions.

• Formally manage architectural risks.
MFESA Task 5)
Create Candidate Architectural Visions

Pitfalls

- The architects engineer only one architectural vision.
- Management provides insufficient resources.
- Management confuses the architectural vision with the completed architecture.
- Management does not permit architects to make mistakes.
- The architects compare the architectural visions prematurely.
- The architects do not compare the pros and cons of the candidate visions.
MFESA Task 6) Analyze Reusable Components and their Sources

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models
Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions

**Task 6) Analyze Reusable Components and their Sources**

Task 7) Select or Create Most Suitable Architectural Vision
Task 8) Complete and Maintain the Architecture
Task 9) Evaluate and Accept the Architecture
Task 10) Ensure Architectural Integrity
MFESA Task 6) Analyze Reusable Components and their Sources

Goal:

- Determine if any existing components are potentially reusable as part of the architecture of the current system or subsystem.

Objectives:

- Identify any existing components that are potentially reusable as part of the architecture of the current system or subsystem.
- Evaluate these components for suitability.
- Evaluate the sources of these components for suitability.
- Provide a set of potentially reusable components to influence (and possibly include in) a set of initial draft architectural models.
MFESA Task 6) Analyze Reusable Components and their Sources

**Inputs:**
- Architectural concerns
- Candidate architectural visions
- Candidate reusable architectural elements
- Documentation (both technical and non-technical) for the candidate reusable components and their sources
- Known architectural risks and opportunities

**Steps:**
1. Identify potentially reusable components and their sources.
2. Characterize the potentially reusable components and their sources.
3. Evaluate the potentially reusable components and their sources.
4. Conditionally select the most suitable reusable components and their sources.
5. Identify any new architectural risks and opportunities

**Outputs:**
- Market surveys
- Potentially reusable architectural components list
- Potentially reusable component descriptions
- New and updated architectural risks and opportunities
MFESA Task 6) Analyze Reusable Components and their Sources

Guidelines

• Use appropriate decision techniques.
• Perform tasks 6 and 7 concurrently.
• Formally manage architectural risks.
MFESA Task 6) Analyze Reusable Components and their Sources

Pitfalls

- Authoritative stakeholders assume reuse will improve cost and schedule.
- Insufficient information exists for evaluation and reuse.
- Stakeholders have an unrealistic expectation of “exact fit.”
- Developers have little or no control over future changes.
- The source organization (e.g., vendor) fails to adequately maintain a reusable architectural component.
- Legal rights are unacceptable.
- Incompatibilities exist with underlying technologies.
MFESA Task 7) Select or Create the Most Suitable Architectural Vision

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models
Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources

Task 7) Select or Create Most Suitable Architectural Vision

Task 8) Complete and Maintain the Architecture
Task 9) Evaluate and Accept the Architecture
Task 10) Ensure Architectural Integrity
MFESA Task 7) Select or Create the Most Suitable Architectural Vision

Goal:

• Obtain a single architectural vision for the system or subsystem architecture from the competing candidate visions.

Objectives:

• Ensure that the selected architectural vision has been properly judged to be most suitable for the system or subsystem architecture.

• Provide a proper foundation on which to complete the engineering of the system or subsystem architecture.
MFESA Task 7) Select or Create the Most Suitable Architectural Vision

Inputs:
- Architectural concerns
- Candidate architectural vision components
- Architectural concern versus vision component matrix
- Competing architectural visions list
- Draft architectural vision documents
- Known architectural risks and opportunities

Steps:
1. Determine the selection criticality.
2. Determine the required selection resources.
3. Determine the evaluation approach.
4. Evaluate the competing candidate architectural visions.
5. Select the most suitable architectural vision.
6. Optionally create the new most suitable architectural vision.
7. Approve the architectural vision.
8. Identify any new architectural risks and opportunities

Outputs:
- Architectural concern versus candidate architectural vision matrix
- Architectural vision selection reports
- Architectural vision document
- New and updated architectural risks and opportunities
### MFESA Task 7) Select or Create the Most Suitable Architectural Vision

<table>
<thead>
<tr>
<th>Architectural Concerns</th>
<th>Architectural Vision 1</th>
<th>Architectural Vision 2</th>
<th>Architectural Vision 3</th>
<th>Architectural Vision 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Development Cost</td>
<td>0</td>
<td>++</td>
<td>--</td>
<td>0</td>
</tr>
<tr>
<td>Development Schedule</td>
<td>+</td>
<td>++</td>
<td>--</td>
<td>-</td>
</tr>
<tr>
<td>Interoperability</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Performance</td>
<td>+</td>
<td>--</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Portability</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>Reliability</td>
<td>+</td>
<td>-</td>
<td>++</td>
<td>-</td>
</tr>
<tr>
<td>Safety</td>
<td>-</td>
<td>-</td>
<td>++</td>
<td>0</td>
</tr>
<tr>
<td>Security</td>
<td>-</td>
<td>++</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>Usability</td>
<td>-</td>
<td>0</td>
<td>-</td>
<td>+</td>
</tr>
</tbody>
</table>
MFESA Task 7) Select or Create the Most Suitable Architectural Vision

Guidelines

• Ensure a commensurate approach.
• Ensure a consistent evaluation approach.
• Ensure complete evaluation criteria.
• Avoid unwarranted assumptions.
• Use common sense when using decision methods to select the most suitable candidate architectural vision.
• Take reuse into account.
• Test reusable architectural component suitability.
• Maintain the architectural vision.
• Formally manage architectural risks.
MFESA Task 7) Select or Create the Most Suitable Architectural Vision

Pitfalls

• Architects use an inappropriate decision method.

• Management provides inadequate decision resources.

• Selecting the most suitable architectural vision is treated as just a technical decision.

• Stakeholders do not understand risks.

• The decision makers are weak.
MFESA Task 8) Complete and Maintain the Architecture

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models
Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources
Task 7) Select or Create Most Suitable Architectural Vision

**Task 8) Complete and Maintain the Architecture**

Task 9) Evaluate and Accept the Architecture
Task 10) Ensure Architectural Integrity
MFESA Task 8) Complete and Maintain the Architecture

Goals:

• Complete system or subsystem architecture based on the selected or created architectural vision.

• Maintain the system or subsystem architecture as the architecturally significant requirements change.

Objectives:

• Complete the interface aspects of the architectural.

• Complete the reuse aspects of the architecture.

• Complete the architectural representations (e.g., architectural models, quality cases, white-papers, and documents).

• Provide a system or subsystem architecture that can be evaluated and accepted by its authoritative stakeholders.
MFESA Task 8)
Complete and Maintain the Architecture

Inputs:
- Architectural concerns
- Incomplete architecture
- Incomplete architectural representations
- Known architectural risks and opportunities

Steps:
1. Complete the draft architectural models of the selected architectural vision.
2. Analyze the architectural models.
3. Complete the quality cases for the architectural focus areas.
4. Complete and document the architectural interfaces.
5. Complete the architectural documentation.
6. Address the remaining architectural reuse issues.
7. Iterate the architecture.
8. Allocate and trace requirements to the architectural elements.
9. Baseline the architectural representations.
10. Identify any new architectural risks and opportunities

Outputs:
- Updated architectural concerns
- Complete and baselined architecture
- Complete and baselined architectural representations
- Requirements trace
- New and updated architectural risks and opportunities
MFESA Task 8)
Complete and Maintain the Architecture

Guidelines

• Address all relevant types of interfaces.

• Maintain the architectural representations to maintain architectural integrity.

• Formally manage architectural risks.
MFESA Task 8)
Complete and Maintain the Architecture

Pitfalls

- Architecture engineering is done.
- Management provides inadequate resources.
- The architectural representations lack configuration control.
- The architecture is not maintained.
- A “beautiful” architecture is frozen solid.
- There is inadequate tool support for architecture maintenance.
MFESA Task 9) Evaluate and Accept the Architecture

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models
Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources
Task 7) Select or Create Most Suitable Architectural Vision
Task 8) Complete and Maintain the Architecture

**Task 9) Evaluate and Accept the Architecture**

Task 10) Ensure Architectural Integrity
MFESA Task 9)
Evaluate and Accept the Architecture

Goals:

- Monitor and determine the quality of the system or subsystem architecture and associated representations.
- Monitor and determine the quality of the process used to engineer the system or subsystem architecture.
- Provide information that can be used to determine the passage or failure of architectural milestones.
- Enable architectural defects, weaknesses, and risks to be fixed and managed before they negatively impact system quality and the success of the system development/enhancement project.
- Accept the system or subsystem architecture based on the results of the evaluations.
MFESA Task 9)
Evaluate and Accept the Architecture

Objectives:

- **Internally verify** the system or subsystem architecture so that architectural
  - Defects are identified and corrected
  - Risks are identified and managed
- **Independently assess** the system or subsystem architecture to determine compliance with architecturally significant product requirements
- **Validate** that the system or subsystem architecture meets the needs of its critical stakeholders
- **Formally review** the system or subsystem architecture by stakeholder representatives at one or more major project reviews
- **Independently evaluate the ‘as performed’ architecture engineering process** to determine compliance with the documented architecture engineering method (for example, as documented in the architecture plan, standards, procedures, and guidance)
MFESA Task 9)
Evaluate and Accept the Architecture

**Inputs:**
- Architectural concerns
- Representations of the current architecture
- Representations of pre-existing architectures
- Architectural patterns
- Known architectural risks and opportunities

**Steps:**
1. Identify architectural concerns that may be implemented via reuse.
2. Identify and analyze the architectural representations of the prior version of the system or subsystem.
3. Identify and analyze the architectural representations of existing variants of system or subsystem.
4. Identify and analyze the architectural representations of any competing systems or subsystems.
5. Identify and analyze the system's product line reference architecture.
6. Identify and analyze the organization's enterprise reference architecture(s).
7. Identify and analyze any industry standard architecture(s).
8. Identify potentially reusable architectural patterns.
9. Identify candidate potentially-reusable architectural elements.
10. Initiate early relationships with potential suppliers of reusable components.
11. Update the architectural concerns.
12. Identify any new architectural risks and opportunities.

**Outputs:**
- Candidate reusable architectural elements
- Updated architectural concerns
- New and updated architectural risks and opportunities
MFESA Task 9)
Evaluate and Accept the Architecture

Tier 1
Assessment Scope
System of Systems

Tier 2
System 1 System 2 System 3 ... System N

Tier 3
Subsystem 1 Subsystem 2 Subsystem 3 ... Subsystem N

Tier 4
Assessment Scope
Segment 1 Segment 2 Segment 3 ... Segment N

Tier 5
Subsegment 1 Subsegment 2 Subsegment 3 ... Subsegment N

Tier 6
Assembly 1 Assembly 2 Assembly 3 ... Assembly N

Tier 7
Subassembly 1 Subassembly 2 Subassembly 3 ... Subassembly N

Tier 8
HW CI1 ... HW CN SW CSC1 ... SW CSCN

Tier 9
HW C1 ... HW CN SW C1 ... SW CN

Tier 10
Part 1 ... Part N SW Unit 1 ... SW Unit N

MFESA Tutorial
Donald Firesmith, 22 April 2009
© 2009 Carnegie Mellon University
MFESA Task 9)
Evaluate and Accept the Architecture

Guidelines

- Use evaluations to support architectural milestones.
- Continuously evaluate the architecture and its representations.
- Internally evaluate models.
- Perform architecture analysis substeps.
- Collaborate with the stakeholders.
- Tailor software evaluation methods.
- Perform independent architecture assessments.
- Formally review the architecture.
- Verify architectural consistency.
- Perform cross-component consistency checking.
- Perform both static and dynamic checking.
- Set the evaluation scope based on risk and available resources.
- Formally manage architectural risks.
MFESA Task 9)
Evaluate and Accept the Architecture

Pitfalls

- Disagreement exists over the need to perform evaluations.
- Consensus does not exist on the evaluation’s scope.
- It is difficult to schedule the evaluations.
- Management provides insufficient evaluation resources.
- There are too few evaluations.
- There are too many evaluations.
- How good is good enough?
- Evaluations are not sufficiently independent.
- The evaluators are inadequate.
- Evaluations only verify the easy concerns.
- The quality cases are poor.
- Stakeholders disagree on the evaluation results.
- The evaluations lack proper acceptance criteria.
- The evaluation results are ignored during acceptance.
- The acceptance package is incomplete.
MFESA Task 10)
Ensure Architectural Integrity

Task 1) Plan and Resource Architecture Engineering Effort
Task 2) Identify the Architectural Drivers
Task 3) Create Initial Architectural Models
Task 4) Identify Opportunities for Reuse of Architectural Elements
Task 5) Create Candidate Architectural Visions
Task 6) Analyze Reusable Components and their Sources
Task 7) Select or Create Most Suitable Architectural Vision
Task 8) Complete and Maintain the Architecture
Task 9) Evaluate and Accept the Architecture

Task 10) Ensure Architectural Integrity
MFESA Task 10)
Ensure Architectural Integrity

Goal:

• Ensure the continued integrity and quality of the system architecture as the system evolves.

Objectives:

• Eliminate inconsistencies within the system architecture and its representations.

• Eliminate inconsistencies between the system architecture and its representations and:
  – Architecturally Significant Requirements
  – Enterprise Architecture(s)
  – Reference Architecture(s)
  – The Design of architectural components
  – The Implementation of architectural components

• The system architecture and its representations do not degrade over time.
MFESA Task 10)
Ensure Architectural Integrity

Inputs:
- System architecture
- System architectural representations
- System change requests...
- Updated work products...
- Known architectural risks and opportunities

Steps:
1. Maintain the architecture and its representations.
2. Determine architectural invariants.
3. Identify changes that threaten architectural integrity.
4. Enforce integrity given changes.
5. Identify any new architectural risks and opportunities.

Outputs:
- Relevant change requests
- Relevant discrepancy reports
- Relevant change control analysis reports
- Updated work products
- New and updated architectural risks and opportunities
MFESA Task 10)
Ensure Architectural Integrity

Guidelines

• Maintain the architectural representations to maintain architectural integrity.
• Consider entire scope of ensure architectural integrity task.
• Consider the sources of architectural change.
• Protect the architectural invariants.
• Determine the scope of architectural integrity.
• Train the architects and designers.
• Formally manage architectural risks.
MFESA Task 10)
Ensure Architectural Integrity

Pitfalls

• The architectural representations become shelfware.
• Architecture engineering is done.
• The architecture is not under configuration management.
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

MFESA Repository of Reusable Method Components
  • Architectural Work Units and Work Products
  • Architectural Workers

MFESA Metamethod

Conclusion
MFESA Repository – Architecture Workers
Architects - Definition

System Architect

the highly specialized role played by a systems engineer when performing system architecture engineering tasks to produce system architecture engineering work products
Types of Architects - Ontology

- Chief/Lead System Architect
- Subsystem Architect
  - System Architect
    - Systems Engineer
  - Software Architect
    - Software Engineer
  - Hardware Architect
    - Hardware Engineer
  
  Engineer
Architects – Primary Responsibilities

Determine and Assess Impact of the Architectural Drivers and Concerns
Develop Architecture and Architectural Representations
Analyze Architecture using Architectural Representations
Evaluate Architecture and Architectural Representations
Maintain Architecture and Architectural Representations
Ensure Architectural Integrity
Architects – Organizational Responsibilities

Lead architectural activities

Manage performance of architecture engineering tasks

Be an architecture advocate

Be a stakeholder advocate

Instantiate and tailor architecture engineering method

Select and acquire architecture engineering tools

Train architecture stakeholders

Evaluate architecture method and process

Interface and collaborate with architecture stakeholders
Architects – Authority

Determine architecture engineering method

Determine architectural work products to produce including models, documents, and architectural prototypes

Select and acquire architecture engineering tools

Determine architecture

Obtain and evaluate Off-The-Shelf architectural components
System Architecture Team - Definition

System Architecture Team

a team responsible for developing and maintaining all or part of a system’s architecture
Types of Architecture Teams - Ontology

- Top-Level Architecture Team
- Specialty Engineering Architecture Teams
- Software Architecture Teams
- Prime Contractor / Integrator Architecture Teams
- Supplier / Vendor Architecture Teams

System of Systems Architecture Team
Subsystem Architecture Teams
Hardware Architecture Teams
Customer Architecture Teams
Subcontractor Architecture Teams

Product Architecture Teams
Reference Architecture Teams

System Architecture Teams

- System Architect
  - Systems Engineer
- Software Architect
  - Software Engineer
- Hardware Architect
  - Hardware Engineer
- Specially Engineer
  - Quality Engineer
- Requirements Engineer
  - Tester
  - Designer
  - Subject Matter Expert
System Architecture Tools - Definition

System Architecture Tool

anything that assists with the production, coordination and maintenance of architectural work products

Many types:

• Whiteboard
• Image Capturing Device
• Word Processor
• Spreadsheet
• General-Purpose Drawing Tool
• Graphical Modeling Tool
• CAD/CAM (Computer Aided Design/Computer Aided Manufacturing)
• Simulation Tool
• Configuration Management Tool
• Requirements Engineering Tool
• Information Architecting Tool
• Business Process Modeling Tool
• Mass/Size/Geometry Modeling Tool
• Software Architecture Tool
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

MFESA Repository of Reusable Method Components
  •  Architectural Work Units and Work Products
  •  Architectural Workers

MFESA Metamethod

Conclusion
MFESA Metamethod - Tasks

Method Selection

- Method Tailoring

Method Needs Assessment

Number of Methods Determination

for each method

Method Reuse Type Determination

Method Reuse

Method Construction

Method Component Selection

Method Component Tailoring

Method Component Integration

Method Documentation

Method Verification

Method Publication
Topics

Motivation

MFESA Overview

MFESA Ontology of Concepts and Terminology

MFESA Metamodel of Reusable Method Components

MFESA Repository of Reusable Method Components
  • Architectural Work Units and Work Products
  • Architectural Workers

MFESA Metamethod

Conclusion
Key Points to Remember

System architecture and system architecture engineering are critical to success.

MFESA is not a system architecture engineering method.

Architectural quality cases make the architects’ case that their architecture sufficiently supports the architecturally significant requirements.

It is critical to capture the rationale for architectural decisions, inventions, and trade-offs.

Architects should keep their work at the right level of abstraction.

Reuse has a major impact on system architecture engineering.

Architecture engineering is never done.
Benefits of using MFESA

The benefits of:

- **Flexibility**: the resulting Architecture Engineering Method meets the unique needs of the stakeholders.

- **Standardization**: built from standard method components implementing best industry practices and based on common terminology and metamodel

Improved system architecture engineering (as-planned) methods and (as-performed) processes.

Improved architectures and architecture representations
Future Informational Website
Questions?

For more information, contact:

Donald Firesmith
Acquisition Support Program (ASP)
Software Engineering Institute (SEI)
dgf@sei.cmu.edu