The *M*ethod-*Framework* for *E*ngineering *S*ystem *A*rchitectures (MFESA)

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Donald Firesmith
12 March 2008

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 system architecture engineering methods and associated processes (process improvement)

MFESA Project

Started January 2007

Collaborators:

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

Work products:

- Reference Book (Auerbach 2008)
- Tutorials and Training Materials
- Articles
- Mapping to Source Documents
- Informational Website

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 – Old Definition

System Architecture

the major components of a system, their relationships, and how they collaborate to meet system requirements

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 achieve quality 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

Architecture	Design
Pervasive (Multiple Components)	*Local* (Single Components)
Strategic Decisions and Inventions	*Tactical* Decisions and Inventions
Higher-Levels of System	*Lower-Levels* of System
Huge Impact on Quality, Cost, & Schedule	*Small Impact* on Quality, Cost, & Schedule
Drives Design and Integration Testing	*Drives* Implementation and Unit Testing
Driven by Requirements and Higher-Level Architecture	*Driven by* Requirements, Architecture, and Higher-Level Design
Mirrors Top-Level Development Team Organization (Conway's Law)	*No* Impact on Top-Level Development Team Organization

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

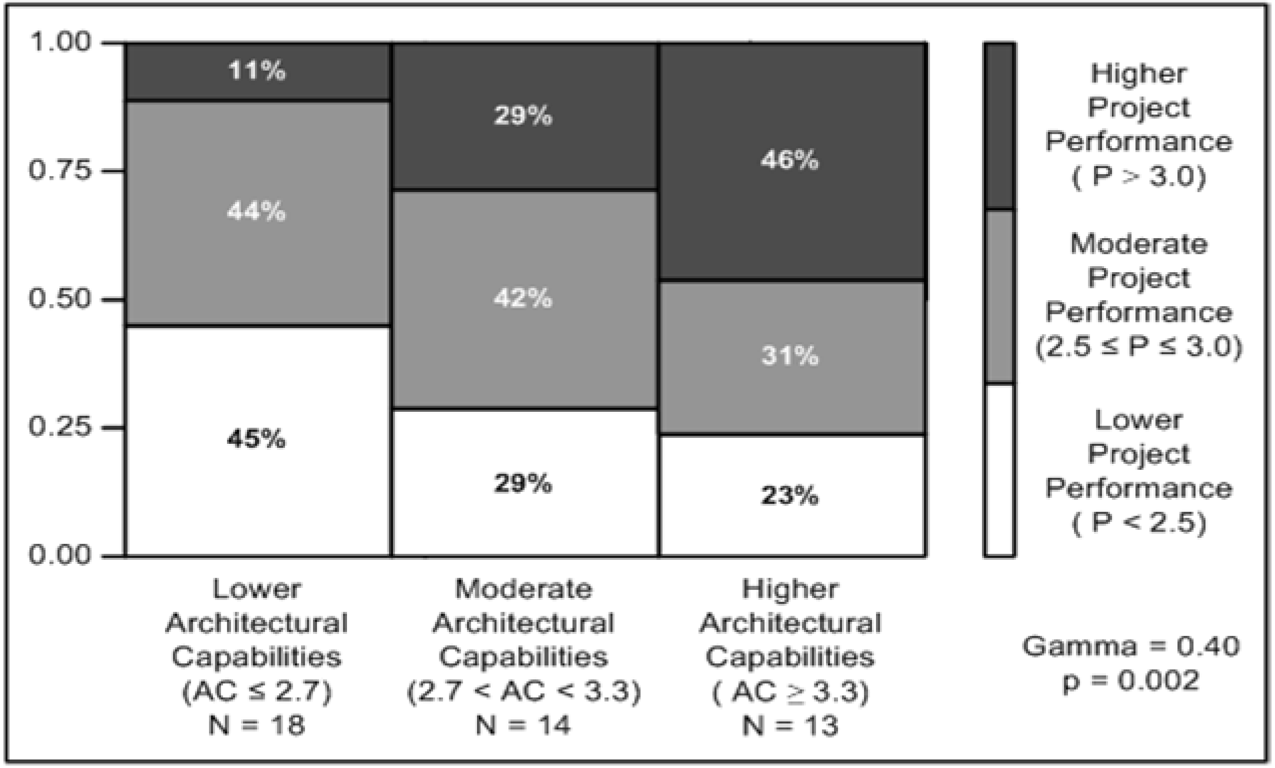
Supports achievement of critical architecturally significant requirements

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



Joe Elm, Dennis R. Goldenson, Khaled El Emam, Nicole Donatelli, and Angelica Neisa, *A Survey of Systems Engineering Effectiveness – Initial Results*, CMU/SEI-2007-SR-014, Software Engineering Institute, November 2007, p. 222.

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.
- 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 use multiple inconsistent architecture engineering methods.

Architecture engineering methods are incompletely documented.

Architects rely too much on architectural engineering tools.

Need for Method Engineering

Systems vary greatly in size, complexity, criticality, domain, technology, operational independence, technological diversity, requirements volatility, required quality characteristics and attributes, and volatility of technology and component parts.

Development organizations vary greatly in degrees of centralization, management culture, engineering culture, expertise, experience, and staff co-location.

Endeavors vary greatly in contracting, type, lifecycle scope, schedule, and funding.

Stakeholders vary in terms of type, numbers, authority, and accessibility.

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

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

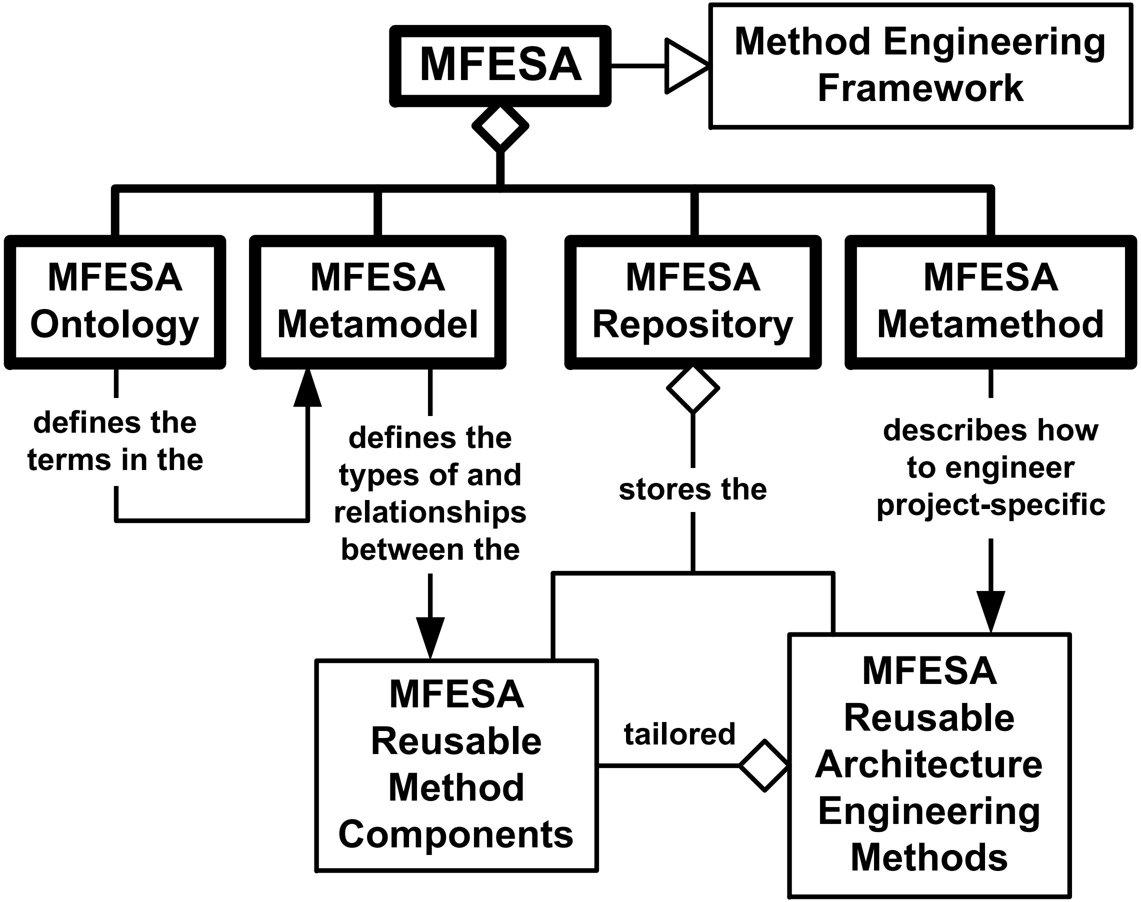
Definition

Method-Framework for Engineering System Architectures (MFESA)

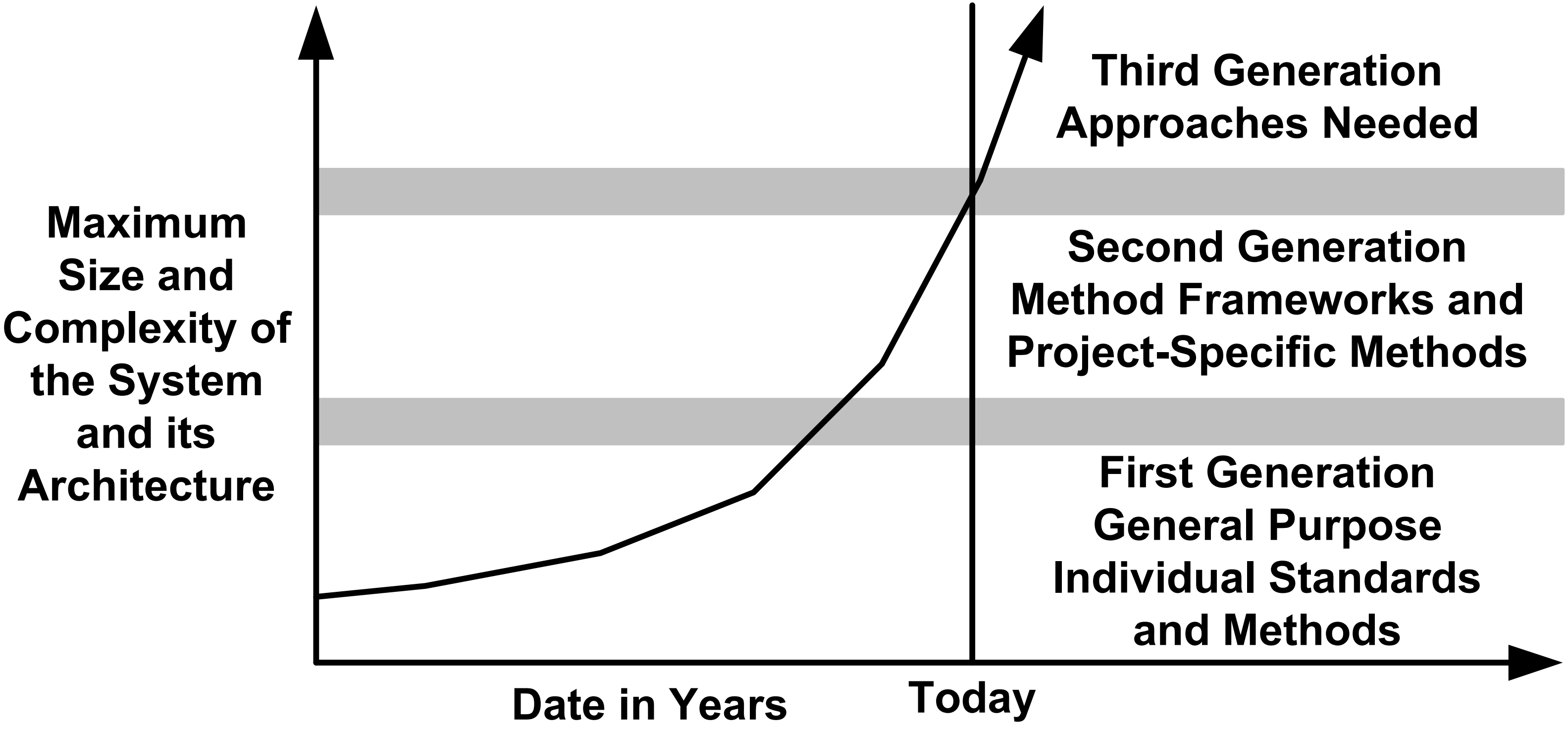
> a method framework for engineering appropriate situation-specific system architecture engineering (SAE) methods

MFESA is <u>not</u> a single system architecture engineering method.

MFESA Components


MFESA
Method Engineering Framework
MFESA Ontology
MFESA Metamodel
MFESA Repository
MFESA Metamethod
defines the terms in the
defines the types of and relationships between the
stores the
describes how to engineer project-specific
MFESA Reusable Method Components
tailored
MFESA Reusable Architecture Engineering Methods

MFESA Addresses Size and Complexity



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 a glossary

Information model of system architecture engineering

Defines foundational 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

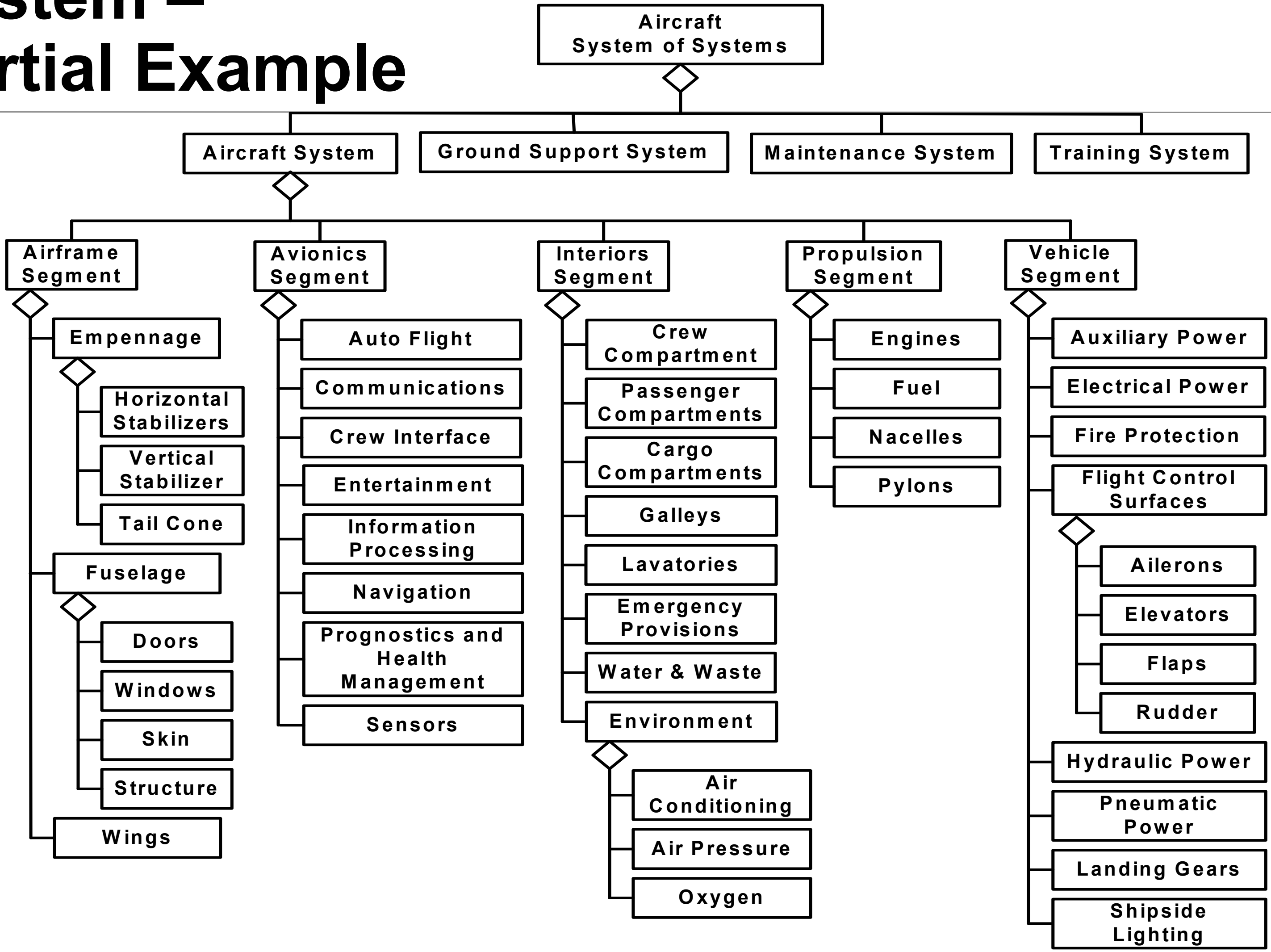
Personnel

Physical interfaces

Software

Tools

System – Partial Example



Some System Characteristics

Multiple Components

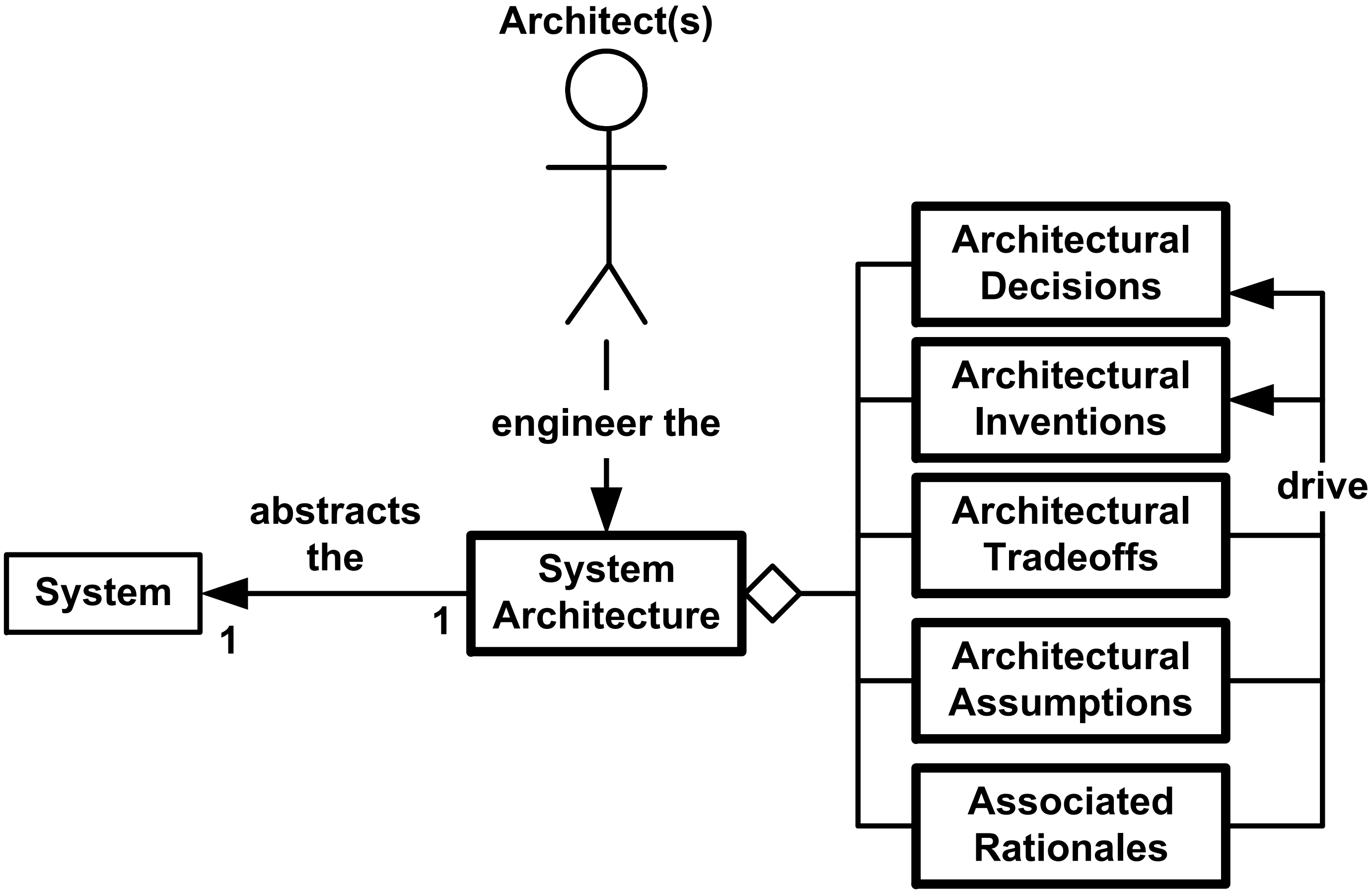
Multiple Interactions between Components

Multiple Structures (Logical and Physical, Static and Dynamic)

Multiple:

- Views and Viewpoints
- Models
- Focus Areas

System Architecture - Ontology



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

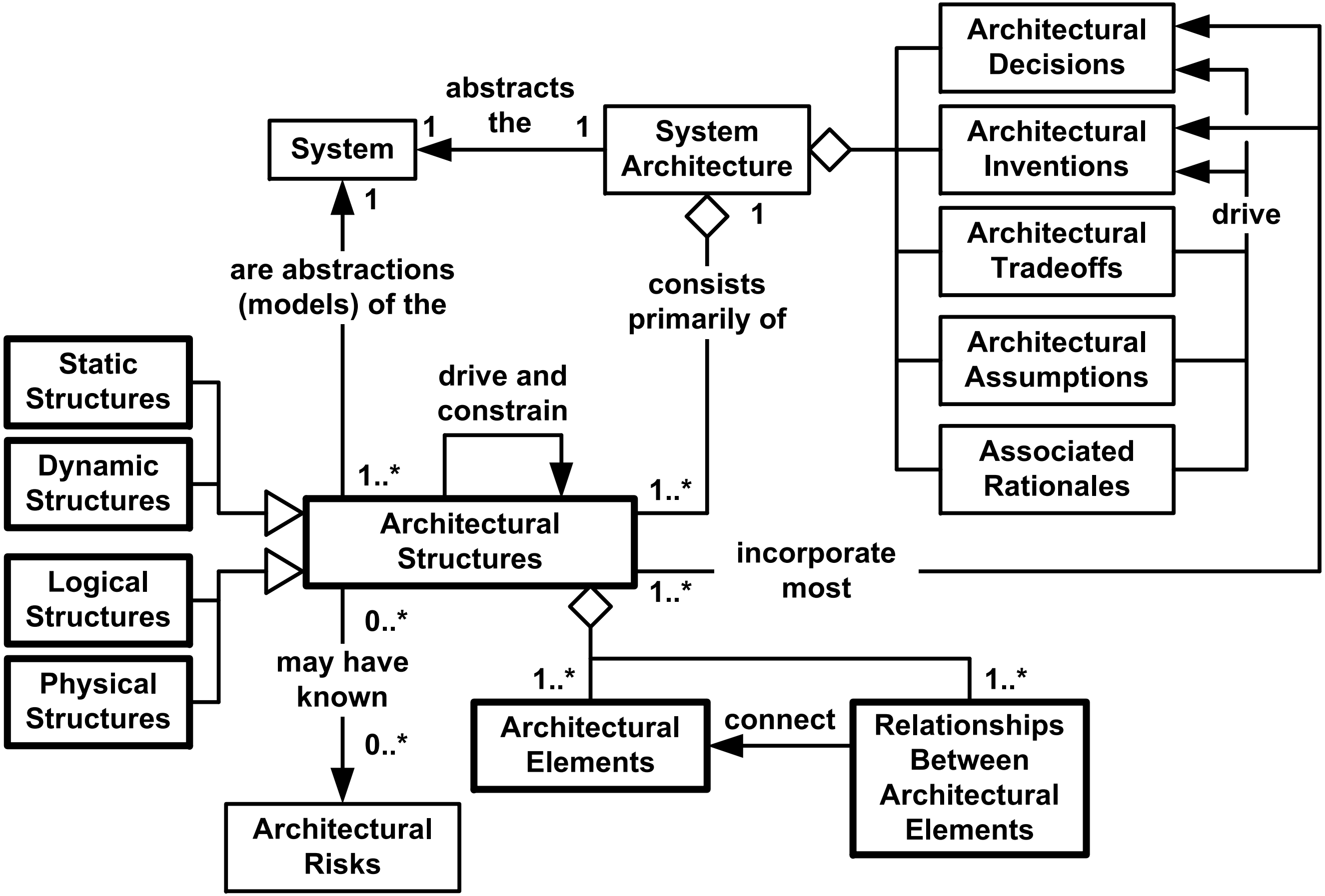
Architectural Element

a part of an architectural structure

Architectural Component

a physical architectural element of a static physical aggregation structure

Architectural Structure - Ontology



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

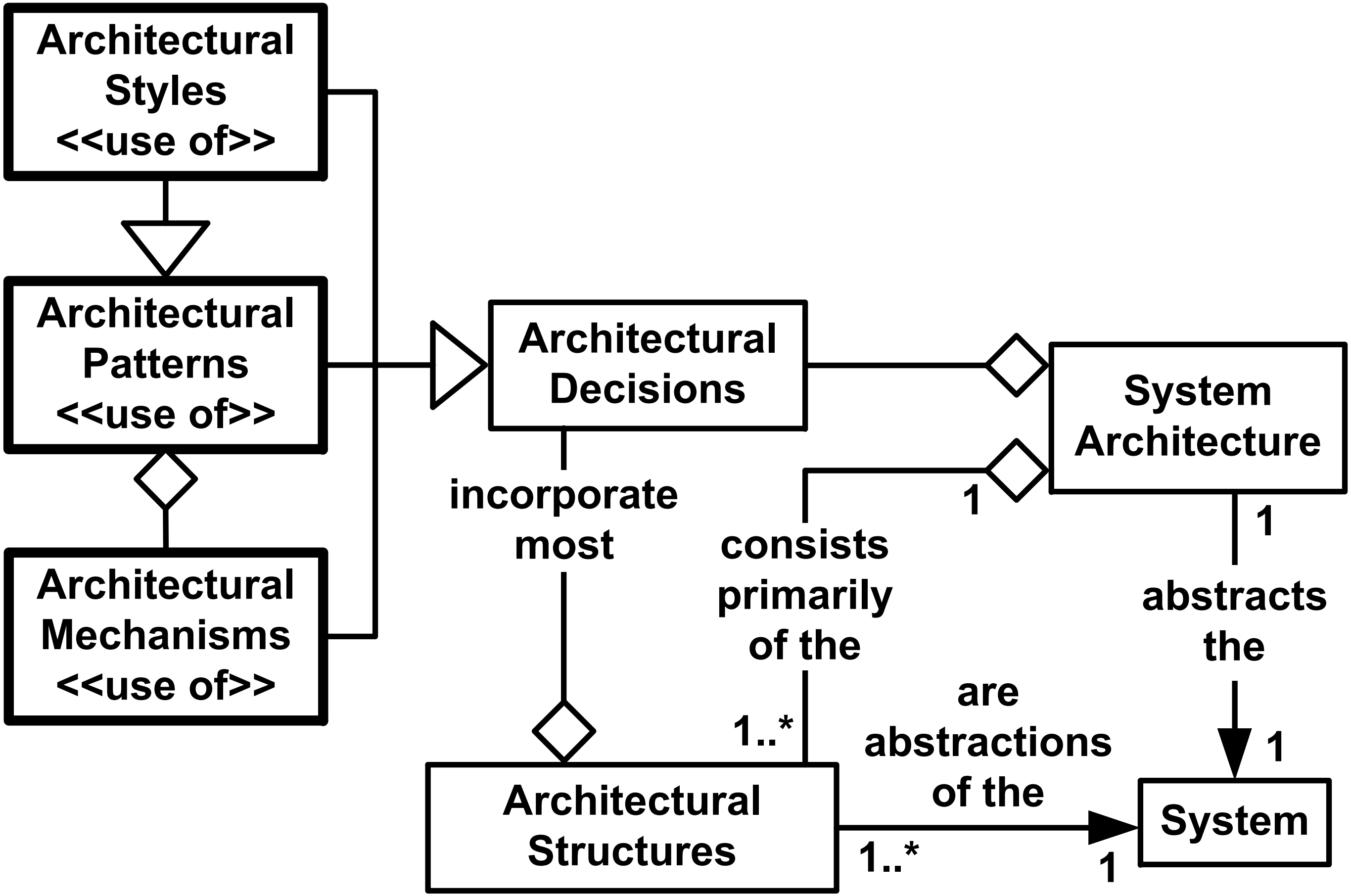
Architectural Style

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 Drivers and Concerns - Definitions

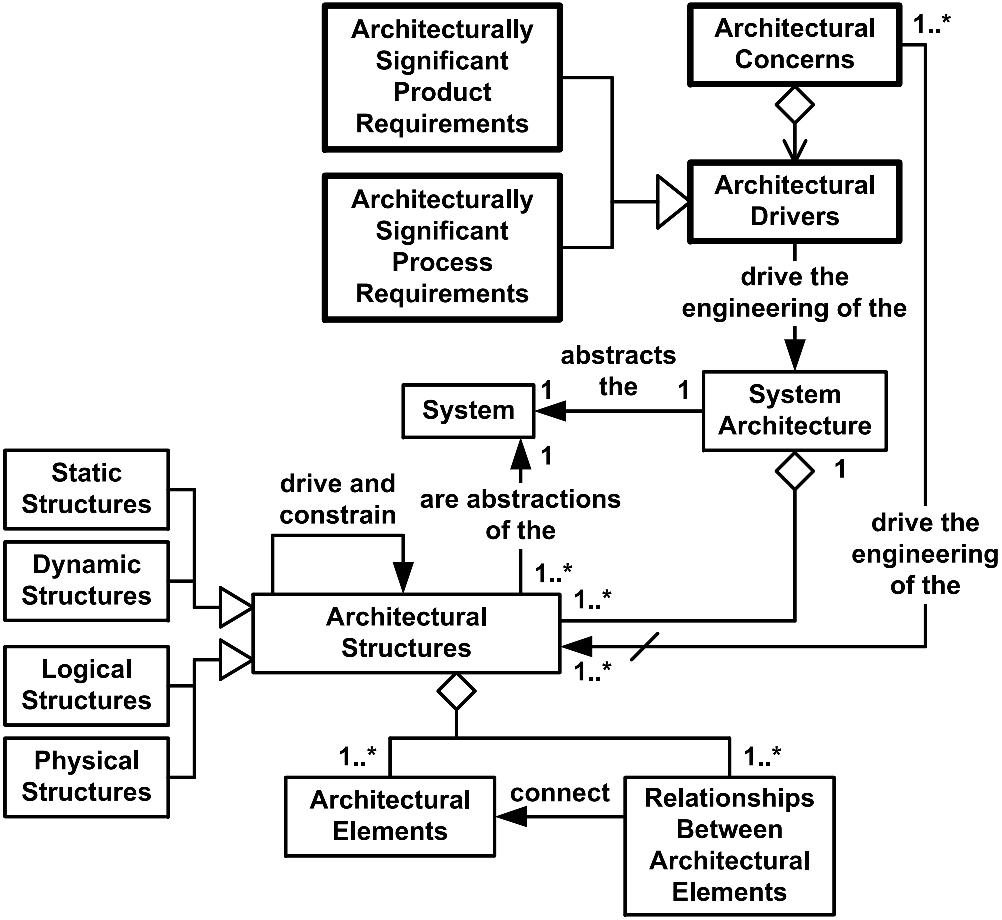
Architectural Driver

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

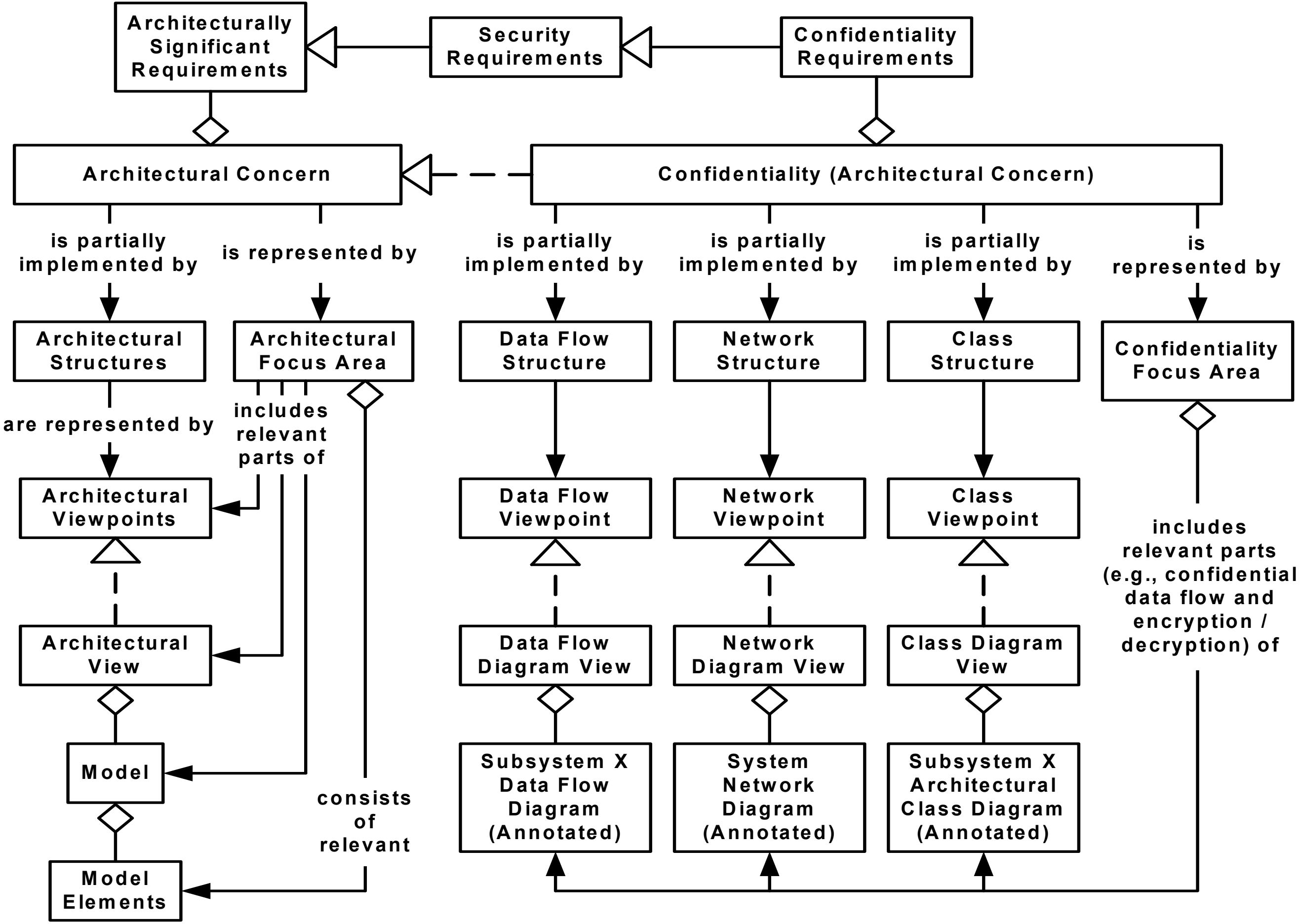
Architectural Concern

a *cohesive* collection of architectural drivers

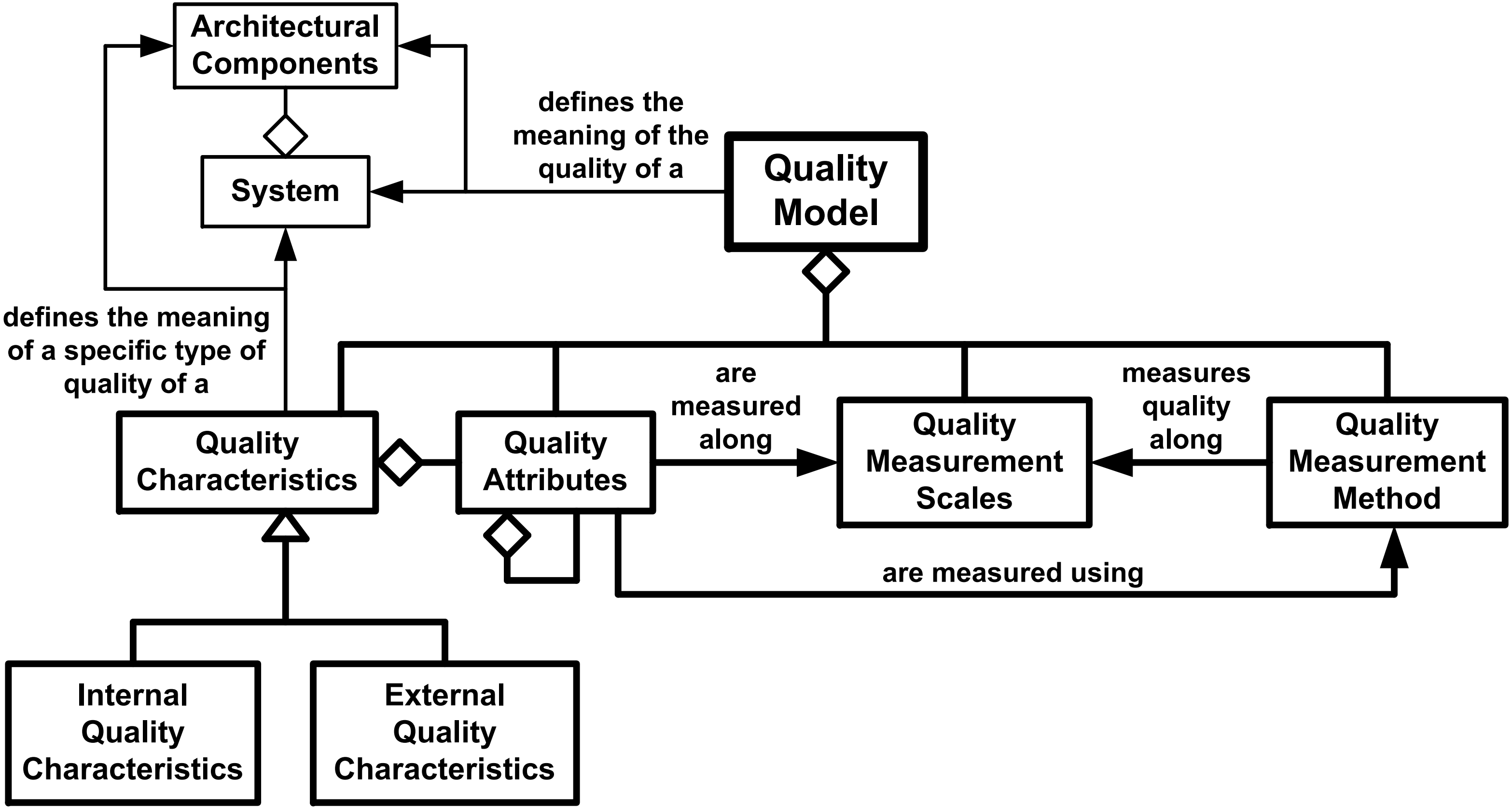
Architectural Drivers and Concerns - Ontology



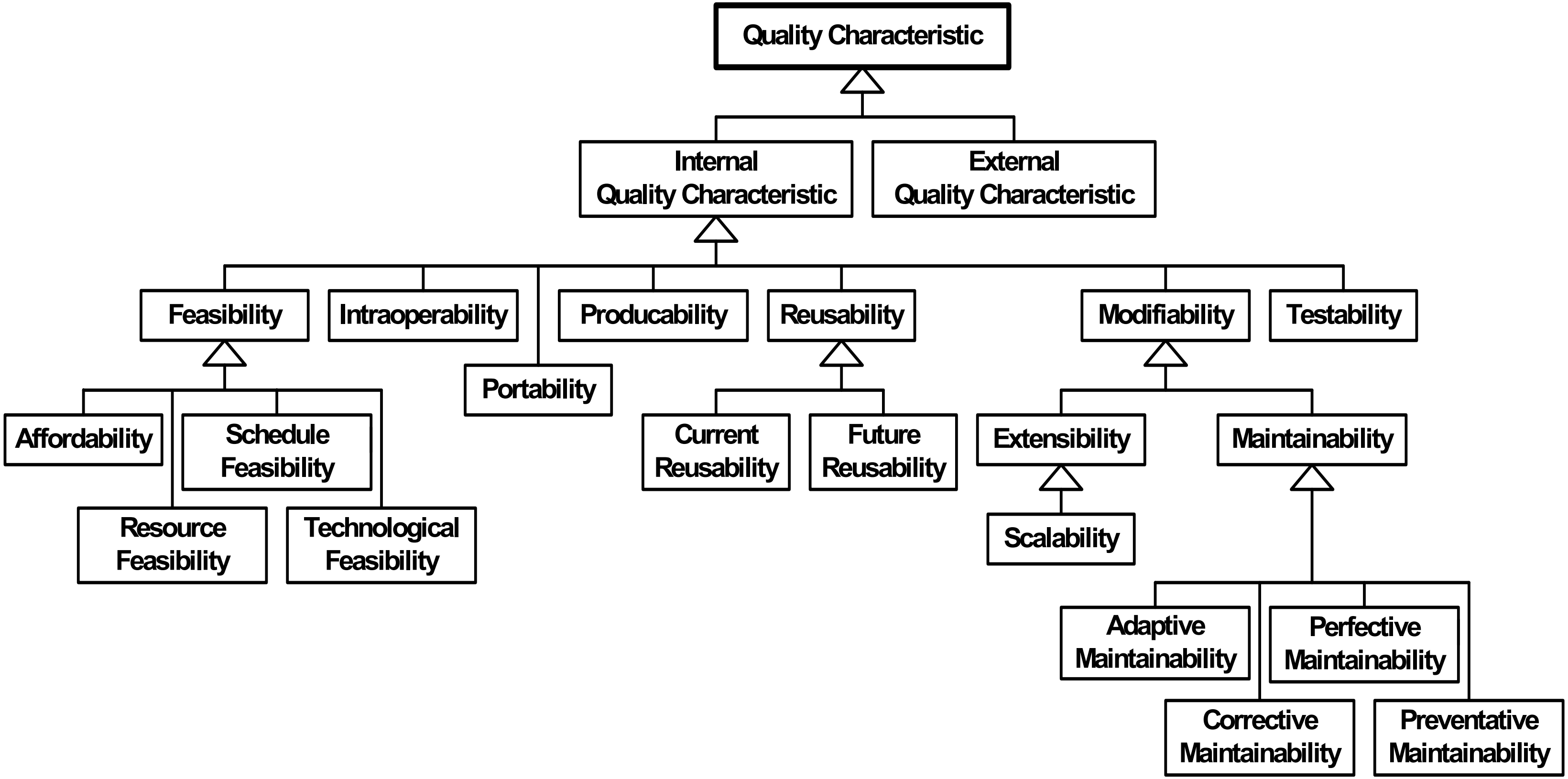
Architectural Concern – An Example


Architecturally Significant Requirements
Security Requirements
Confidentiality Requirements
Architectural Concern
Confidentiality (Architectural Concern)
is partially implemented by
is represented by
is partially implemented by
is partially implemented by
is partially implemented by
is represented by
Architectural Structures
Architectural Focus Area
Data Flow Structure
Network Structure
Class Structure
Confidentiality Focus Area
are represented by
includes relevant parts of
Architectural Viewpoints
Data Flow Viewpoint
Network Viewpoint
Class Viewpoint
includes relevant parts (e.g., confidential data flow and encryption / decryption) of
Architectural View
Data Flow Diagram View
Network Diagram View
Class Diagram View
Model
Subsystem X Data Flow Diagram (Annotated)
System Network Diagram (Annotated)
Subsystem X Architectural Class Diagram (Annotated)
consists of relevant
Model Elements

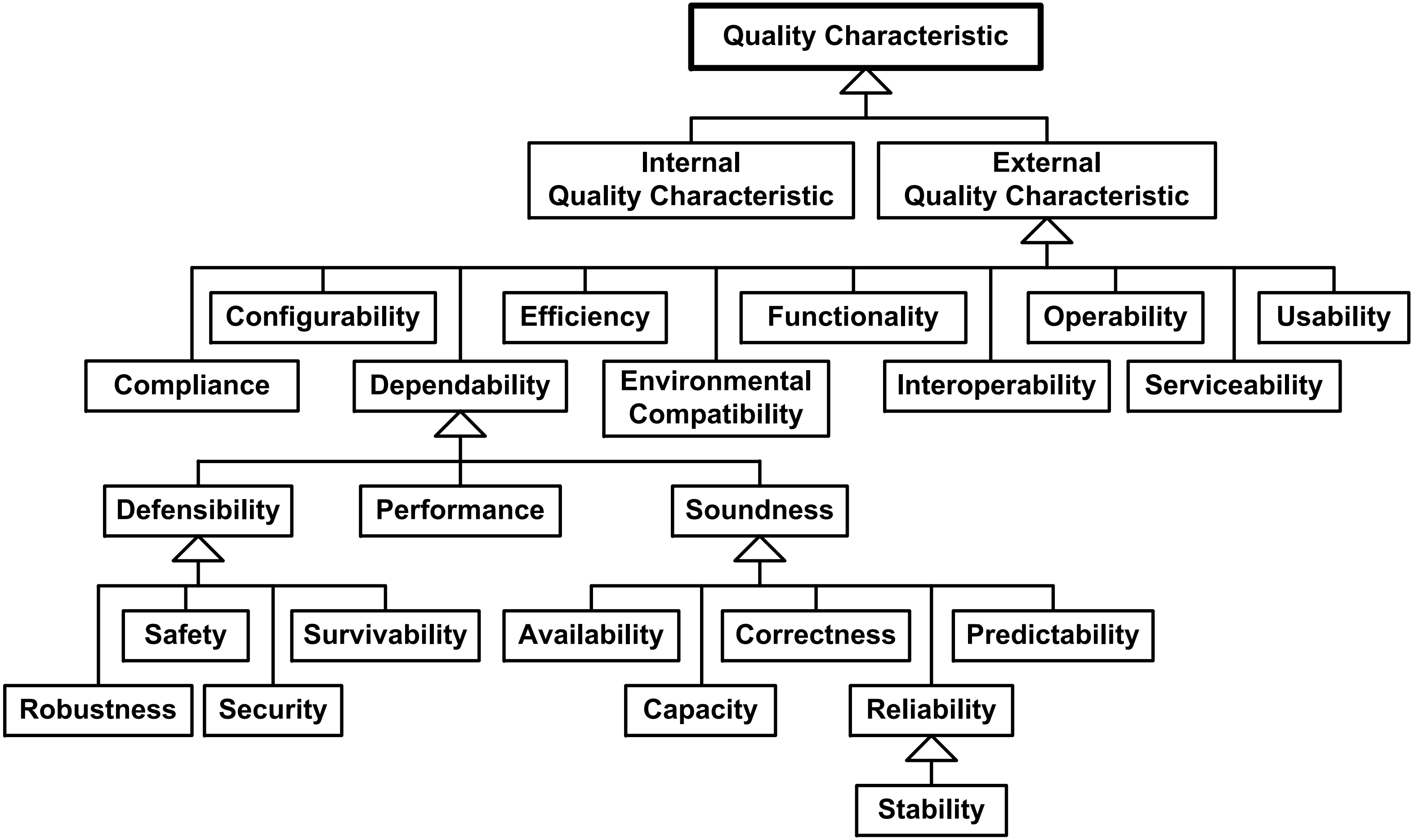
MFESA Quality Model


Architectural Components
System
defines the meaning of the quality of a
Quality Model
defines the meaning of a specific type of quality of a
Quality Characteristics
Quality Attributes
are measured along
Quality Measurement Scales
measures quality along
Quality Measurement Method
are measured using
Internal Quality Characteristics
External Quality Characteristics

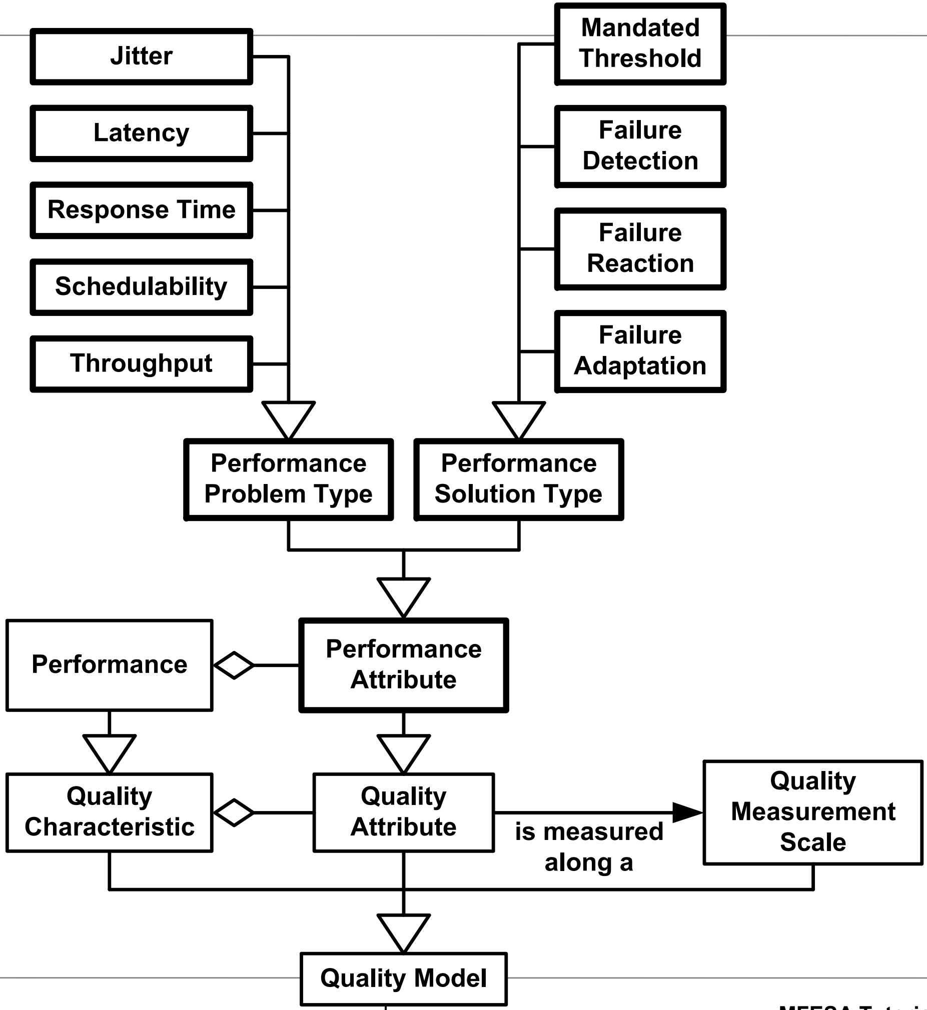
Internal Quality Characteristics



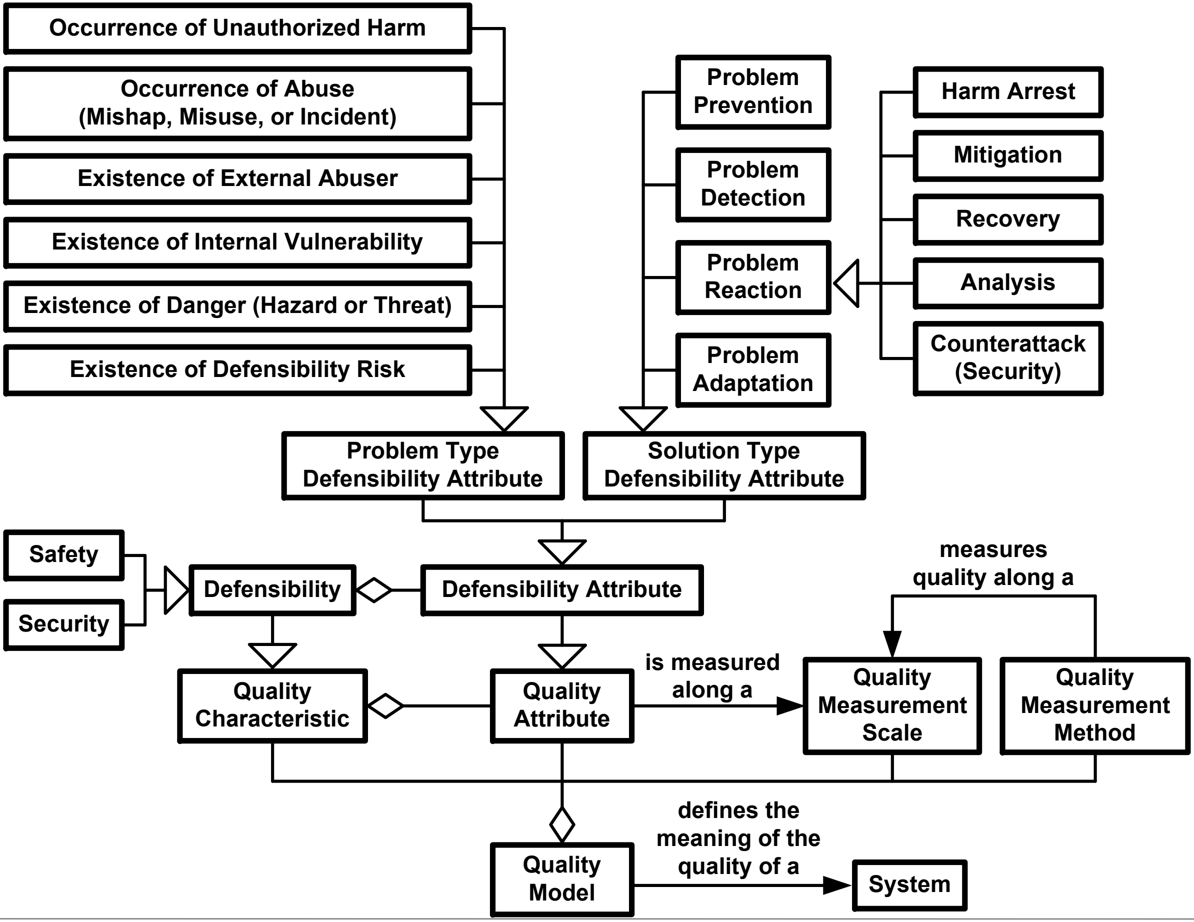
External Quality Characteristics



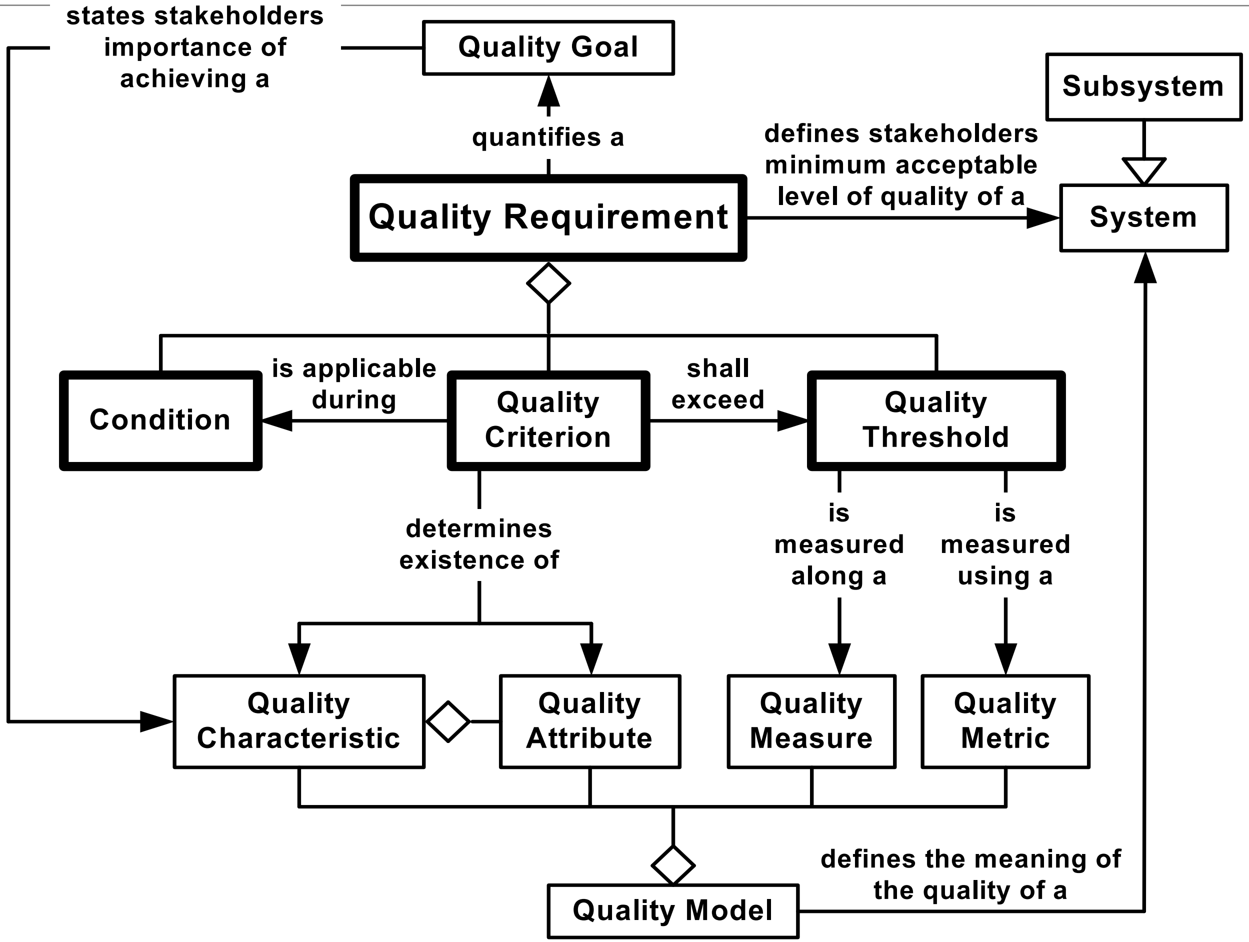
Example Characteristic and Attributes



Example Characteristic and Attributes


Occurrence of Unauthorized Harm
Occurrence of Abuse (Mishap, Misuse, or Incident)
Existence of External Abuser
Existence of Internal Vulnerability
Existence of Danger (Hazard or Threat)
Existence of Defensibility Risk
Problem Prevention
Problem Detection
Problem Reaction
Problem Adaptation
Harm Arrest
Mitigation
Recovery
Analysis
Counterattack (Security)
Problem Type Defensibility Attribute
Solution Type Defensibility Attribute
Safety
Security
Defensibility
Defensibility Attribute
measures quality along a
Quality Characteristic
Quality Attribute
is measured along a
Quality Measurement Scale
Quality Measurement Method
Quality Model
defines the meaning of the quality of a
System

Quality Requirements



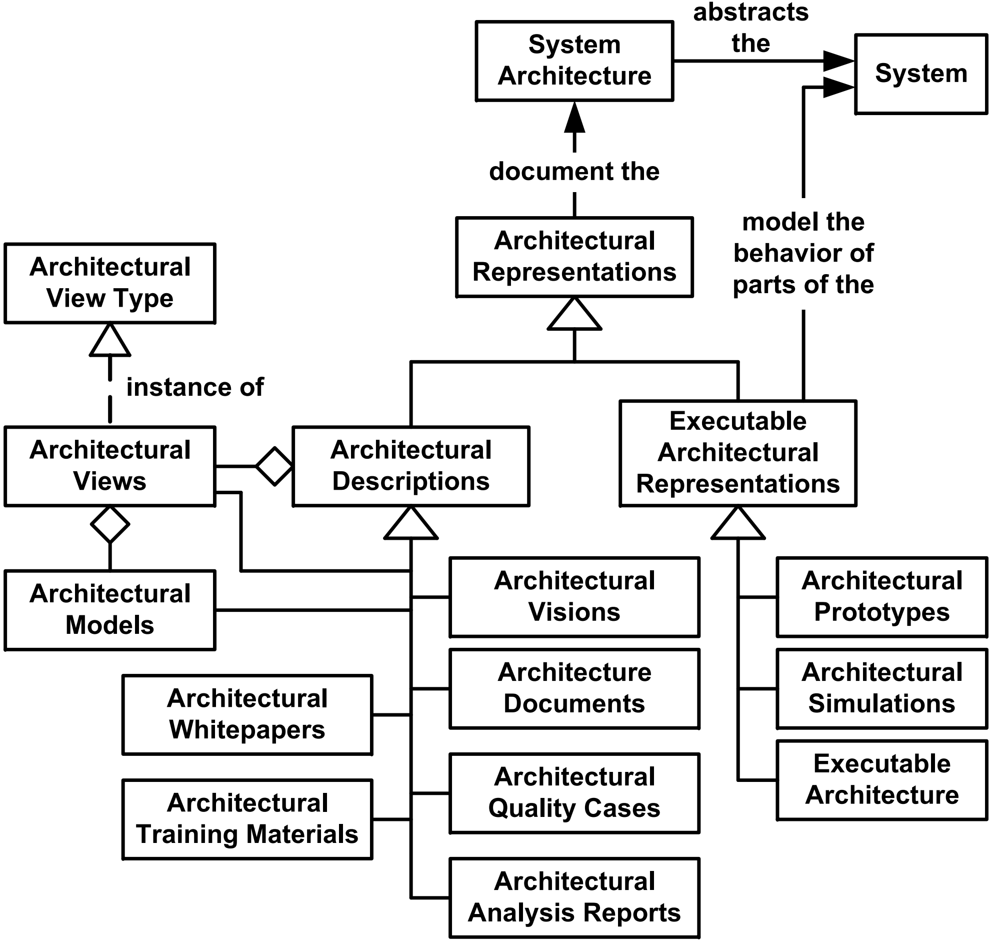
Architectural Representations - Definition

Architectural Representation

a cohesive collection of information that documents a system architecture

Not the same thing as the architecture

Architectural Representations - Ontology


System Architecture
abstracts the
System
document the
Architectural Representations
model the behavior of parts of the
Architectural View Type
instance of
Architectural Views
Architectural Descriptions
Executable Architectural Representations
Architectural Models
Architectural Visions
Architecture Documents
Architectural Whitepapers
Architectural Training Materials
Architectural Quality Cases
Architectural Analysis Reports
Architectural Prototypes
Architectural Simulations
Executable Architecture

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

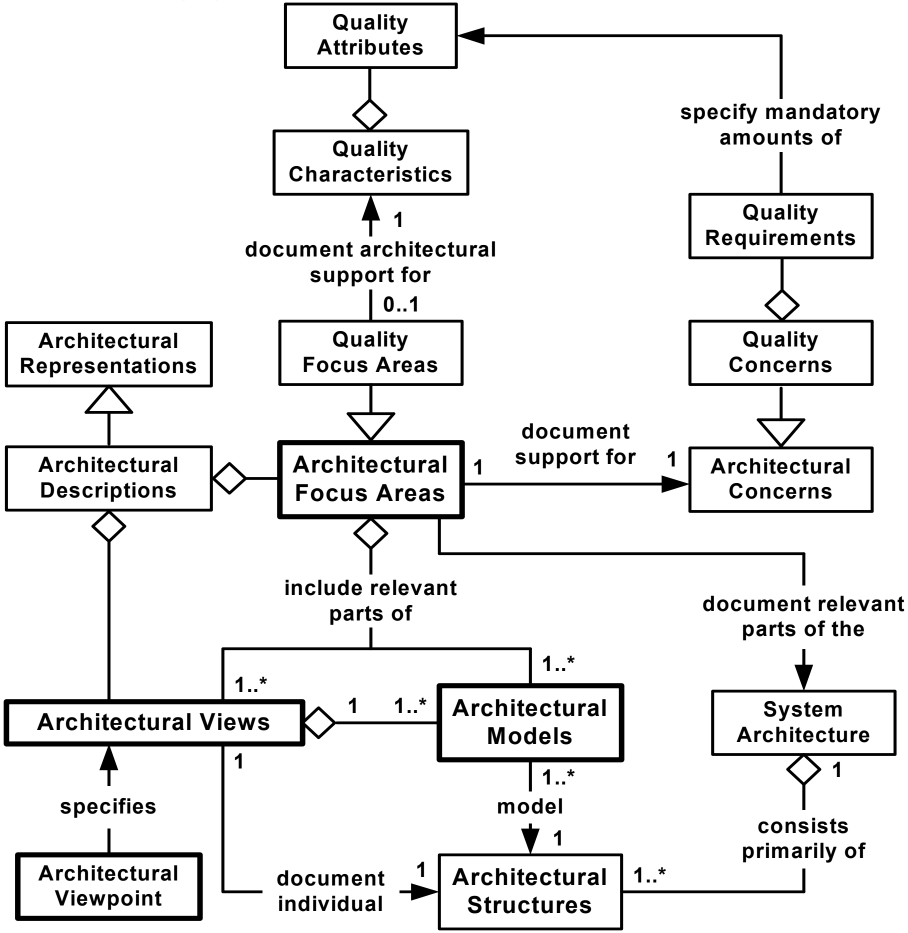
Architectural View

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

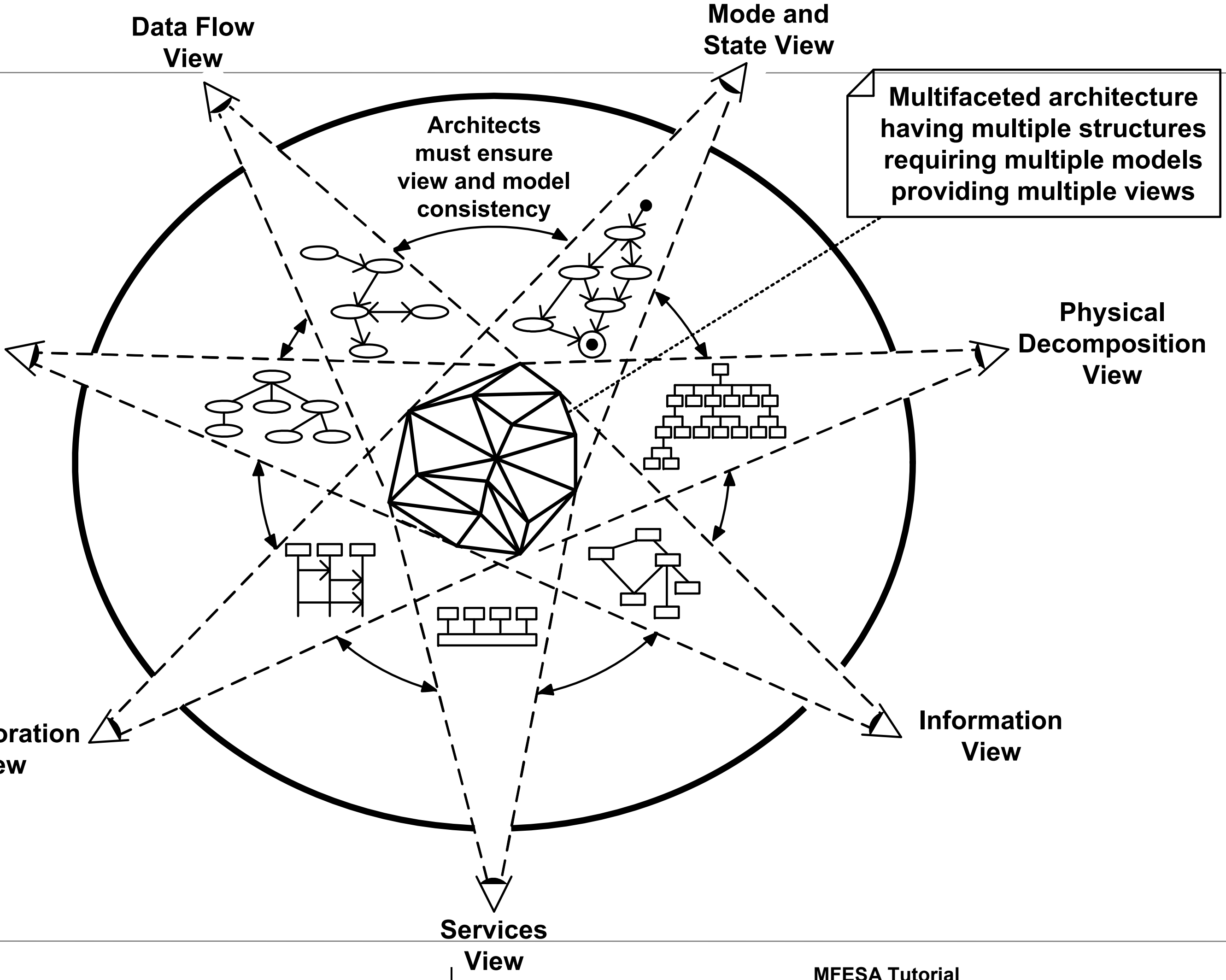
Architectural Focus Area

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 Views



Data Flow View

Mode and State View

Multifaceted architecture having multiple structures requiring multiple models providing multiple views

Architects must ensure view and model consistency

Logical Functional Decomposition View

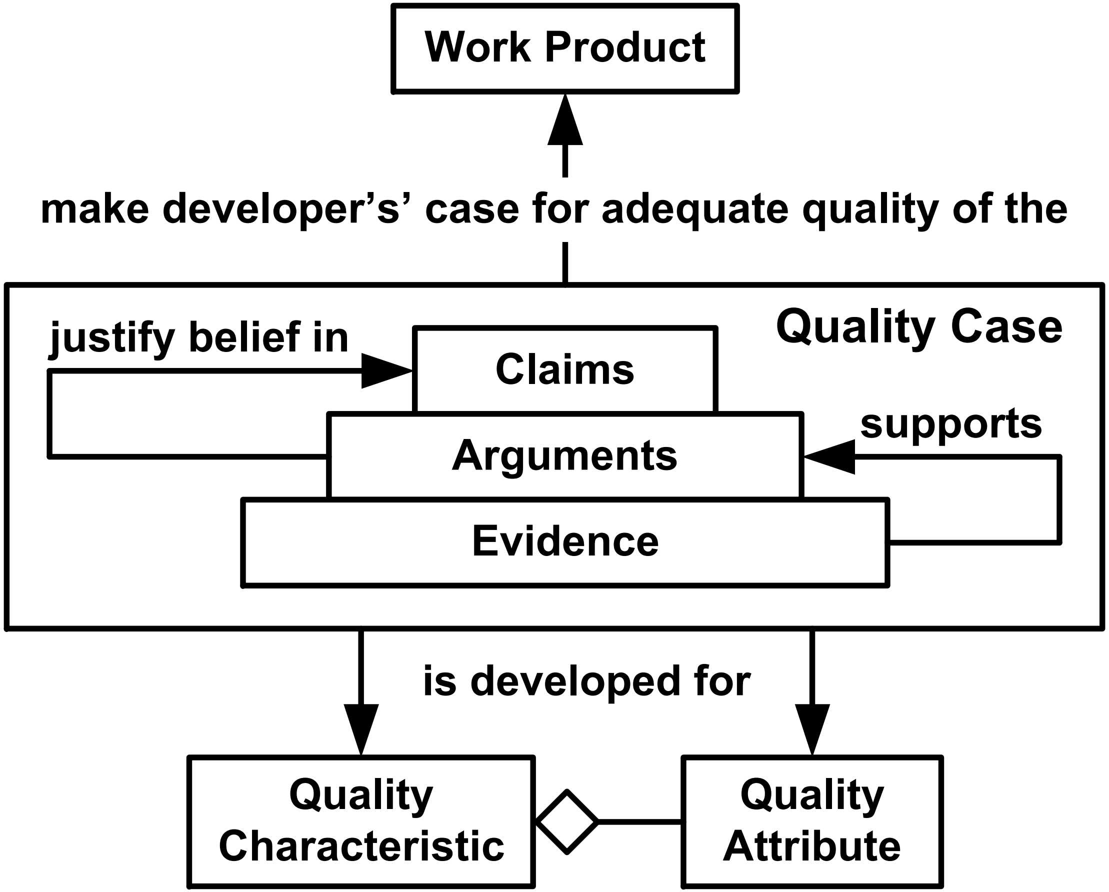
Physical Decomposition View

Collaboration View

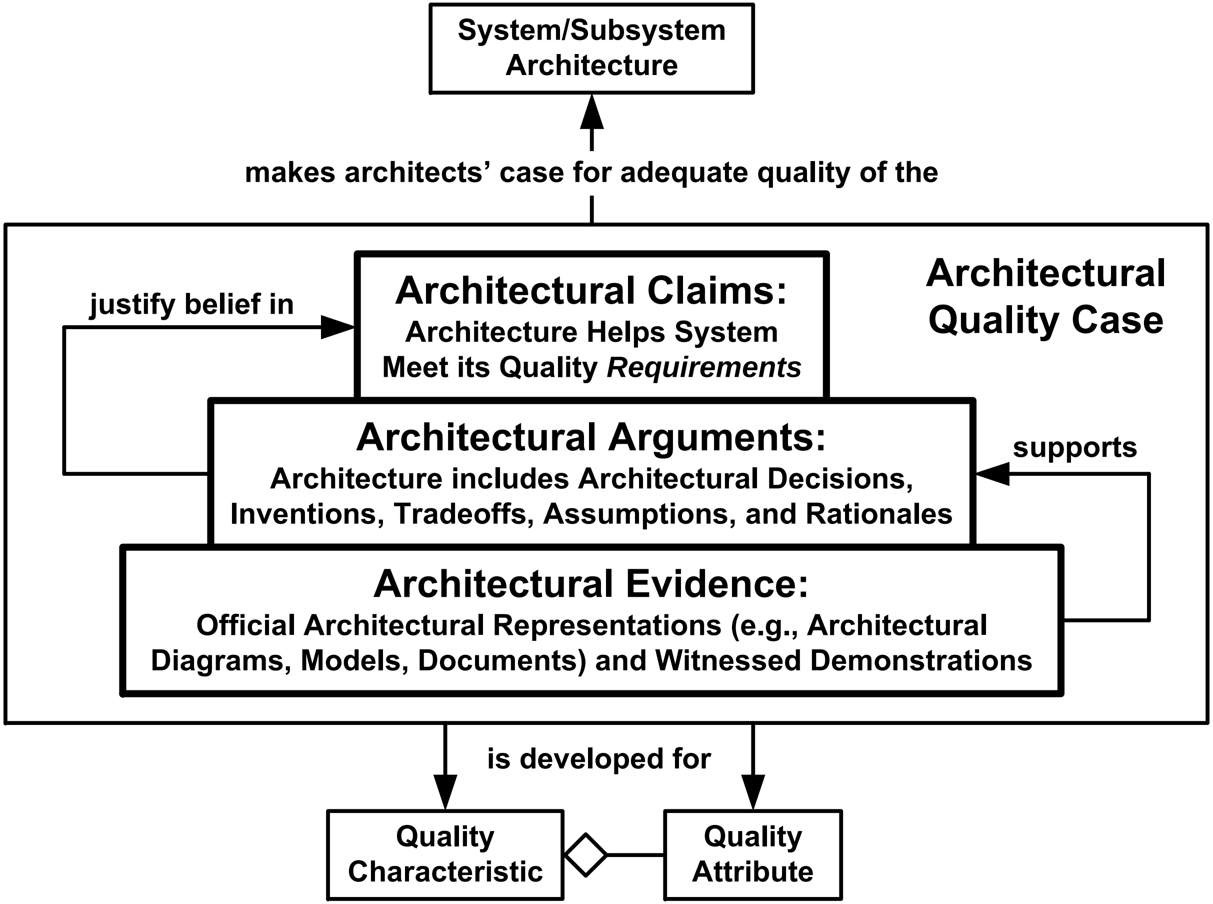
Information View

Services View

Quality Cases



Architectural Quality Cases



System/Subsystem Architecture

makes architects' case for adequate quality of the

Architectural Quality Case

justify belief in

Architectural Claims:
Architecture Helps System Meet its Quality *Requirements*

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

supports

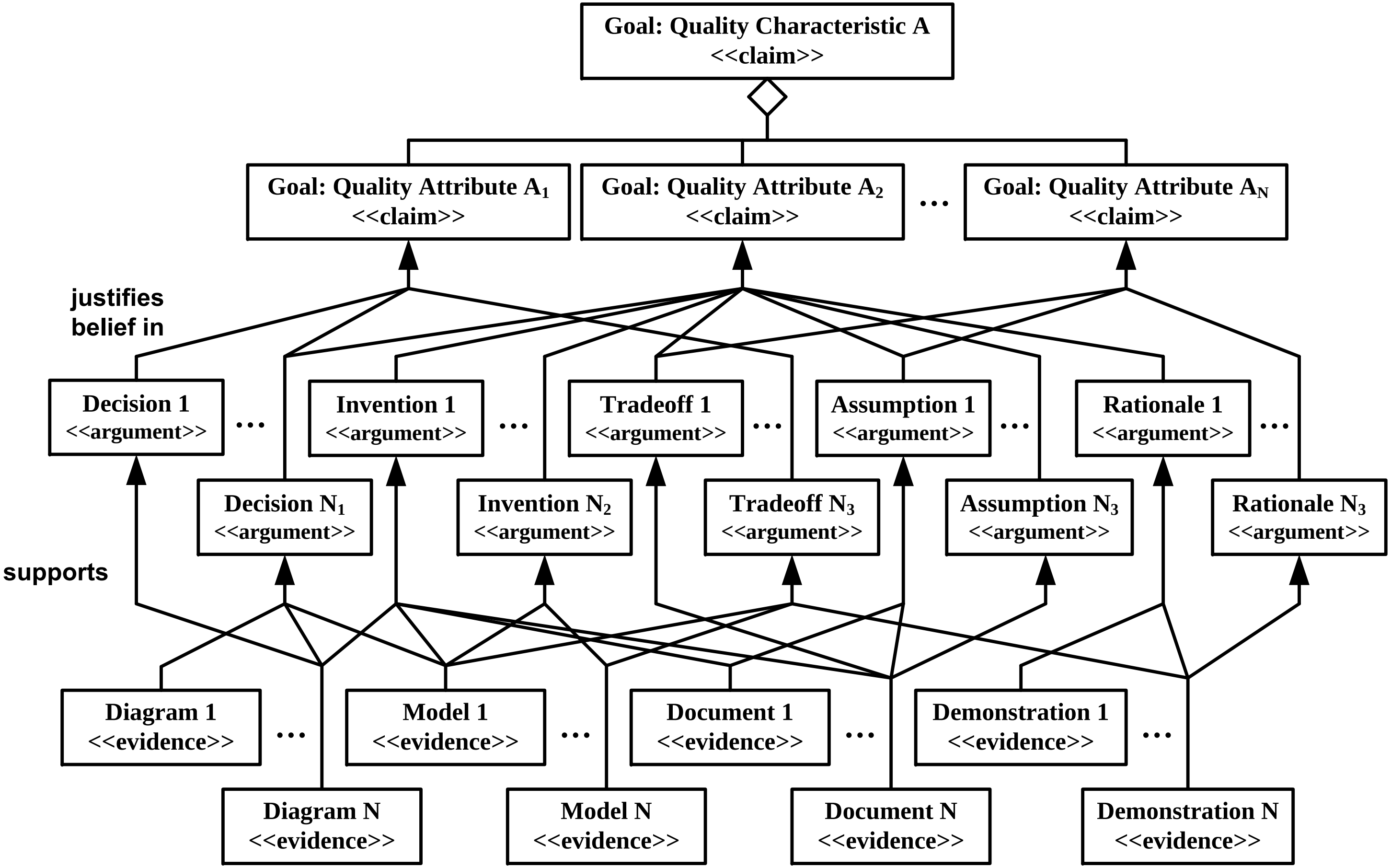
Architectural Evidence:
Official Architectural Representations (e.g., Architectural Diagrams, Models, Documents) and Witnessed Demonstrations

is developed for

Quality Characteristic

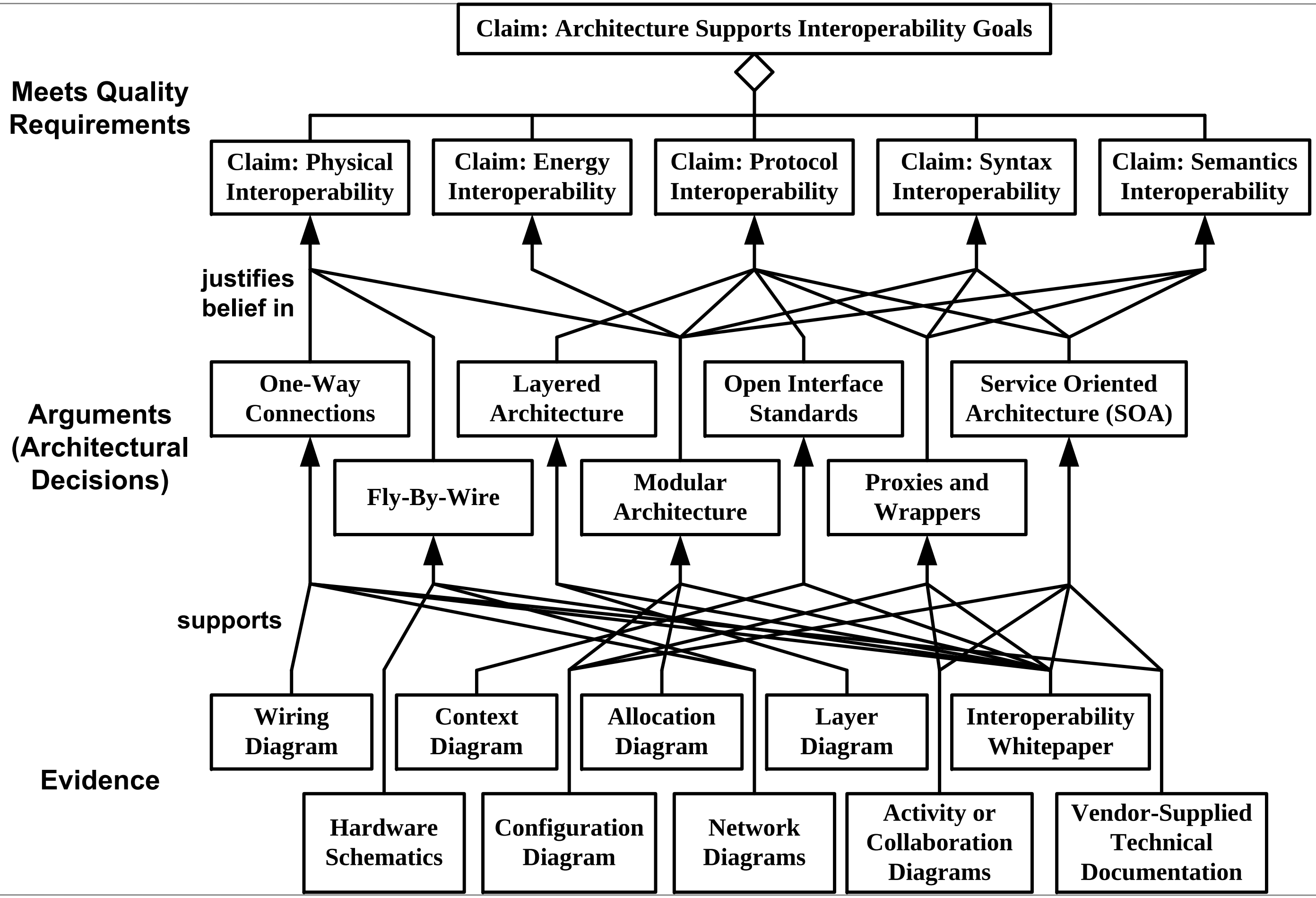
Quality Attribute

Architectural Quality Case Diagram



Goal: Quality Characteristic A <<claim>>
Goal: Quality Attribute A_1 <<claim>>
Goal: Quality Attribute A_2 <<claim>>
…
Goal: Quality Attribute A_N <<claim>>
justifies belief in
Decision 1 <<argument>>
…
Invention 1 <<argument>>
…
Tradeoff 1 <<argument>>
…
Assumption 1 <<argument>>
…
Rationale 1 <<argument>>
…
Decision N_1 <<argument>>
Invention N_2 <<argument>>
Tradeoff N_3 <<argument>>
Assumption N_3 <<argument>>
Rationale N_3 <<argument>>
supports
Diagram 1 <<evidence>>
…
Model 1 <<evidence>>
…
Document 1 <<evidence>>
…
Demonstration 1 <<evidence>>
…
Diagram N <<evidence>>
Model N <<evidence>>
Document N <<evidence>>
Demonstration N <<evidence>>

Example Architectural Quality Case Diagram


Claim: Architecture Supports Interoperability Goals
Meets Quality Requirements
Claim: Physical Interoperability
Claim: Energy Interoperability
Claim: Protocol Interoperability
Claim: Syntax Interoperability
Claim: Semantics Interoperability
justifies belief in
Arguments (Architectural Decisions)
One-Way Connections
Layered Architecture
Open Interface Standards
Service Oriented Architecture (SOA)
Fly-By-Wire
Modular Architecture
Proxies and Wrappers
supports
Evidence
Wiring Diagram
Context Diagram
Allocation Diagram
Layer Diagram
Interoperability Whitepaper
Hardware Schematics
Configuration Diagram
Network Diagrams
Activity or Collaboration Diagrams
Vendor-Supplied Technical Documentation

Architecture Visions and Vision Components - Definitions

Architectural Vision

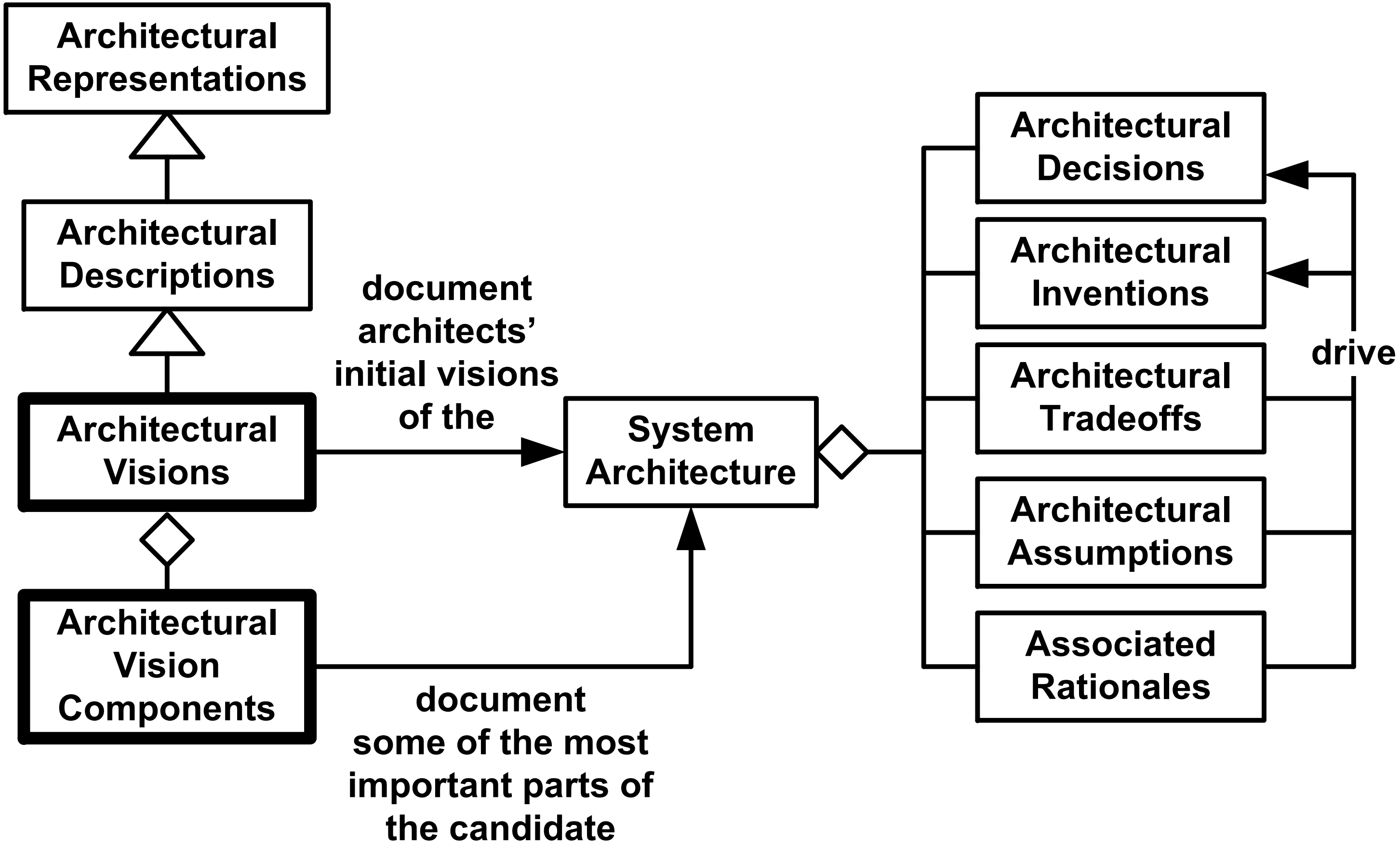
one of the more important actual or potential architectural decisions, inventions, or tradeoffs addressing one or more architectural concerns

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
System Architecture
document some of the most important parts of the candidate
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

MFESA Metamodel

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 24744 (2007-02-15) *Information Technology – Software Engineering – Metamodel for Development Methodologies*

Not based on OMG Metamodel.

System Architecture Engineering – Methods and Processes

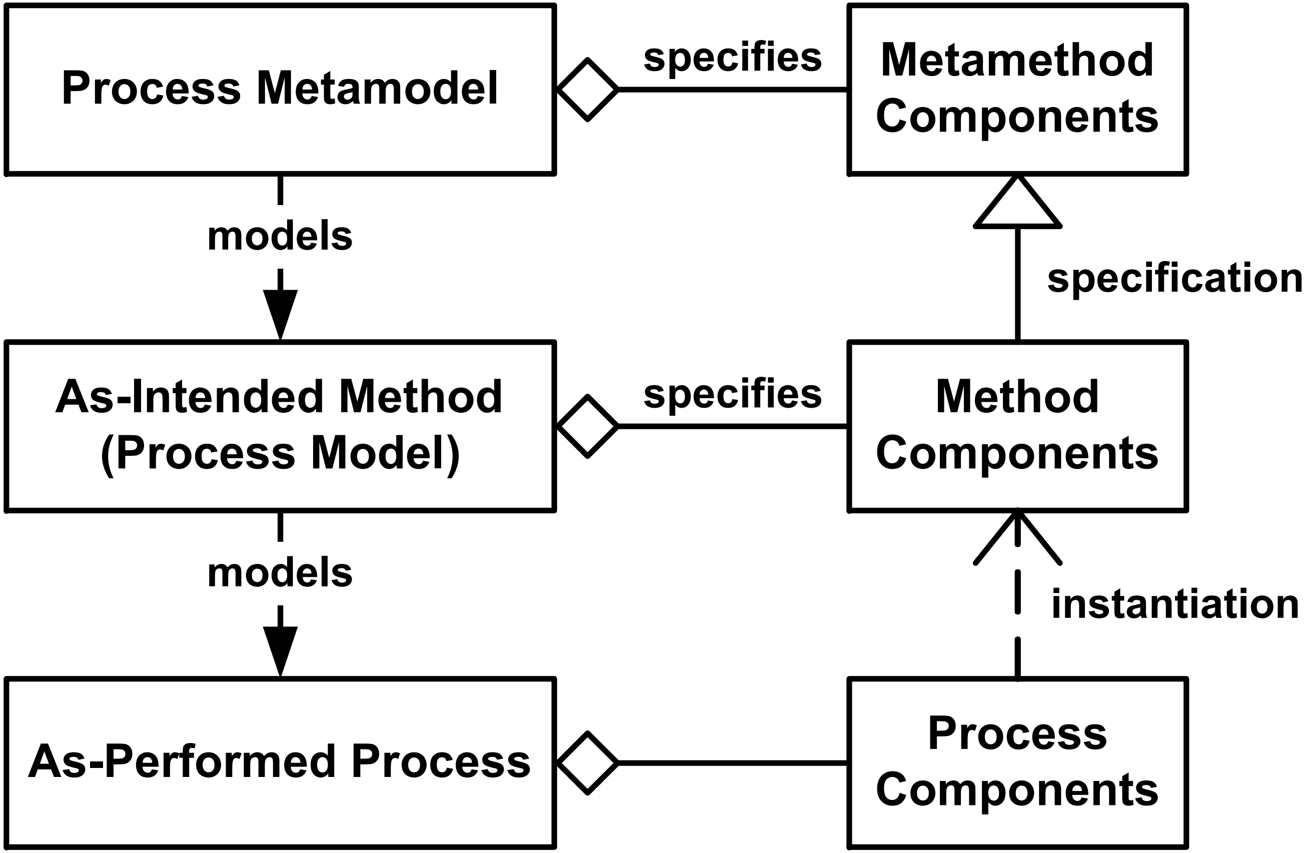
System Architecture Engineering Method

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

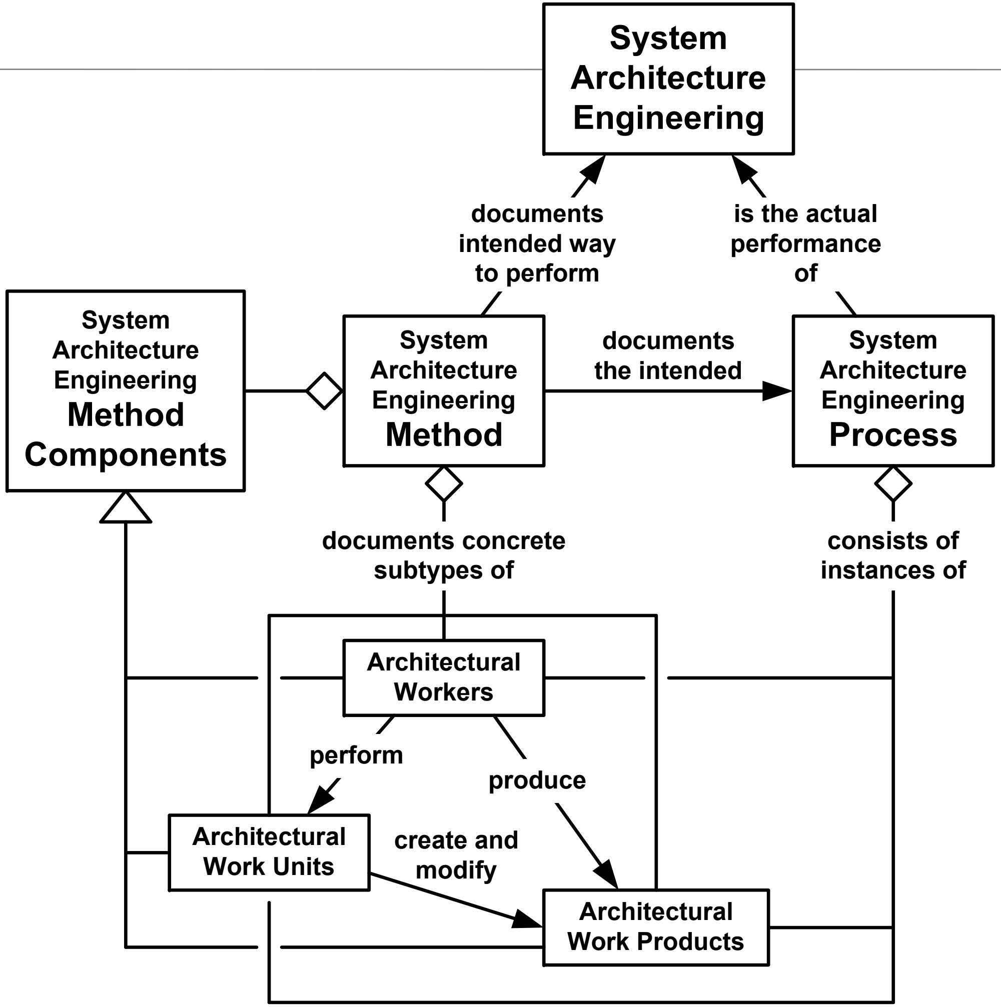
System Architecture Engineering Process

an *actual* way that system architecture engineering is performed in practice on an endeavor

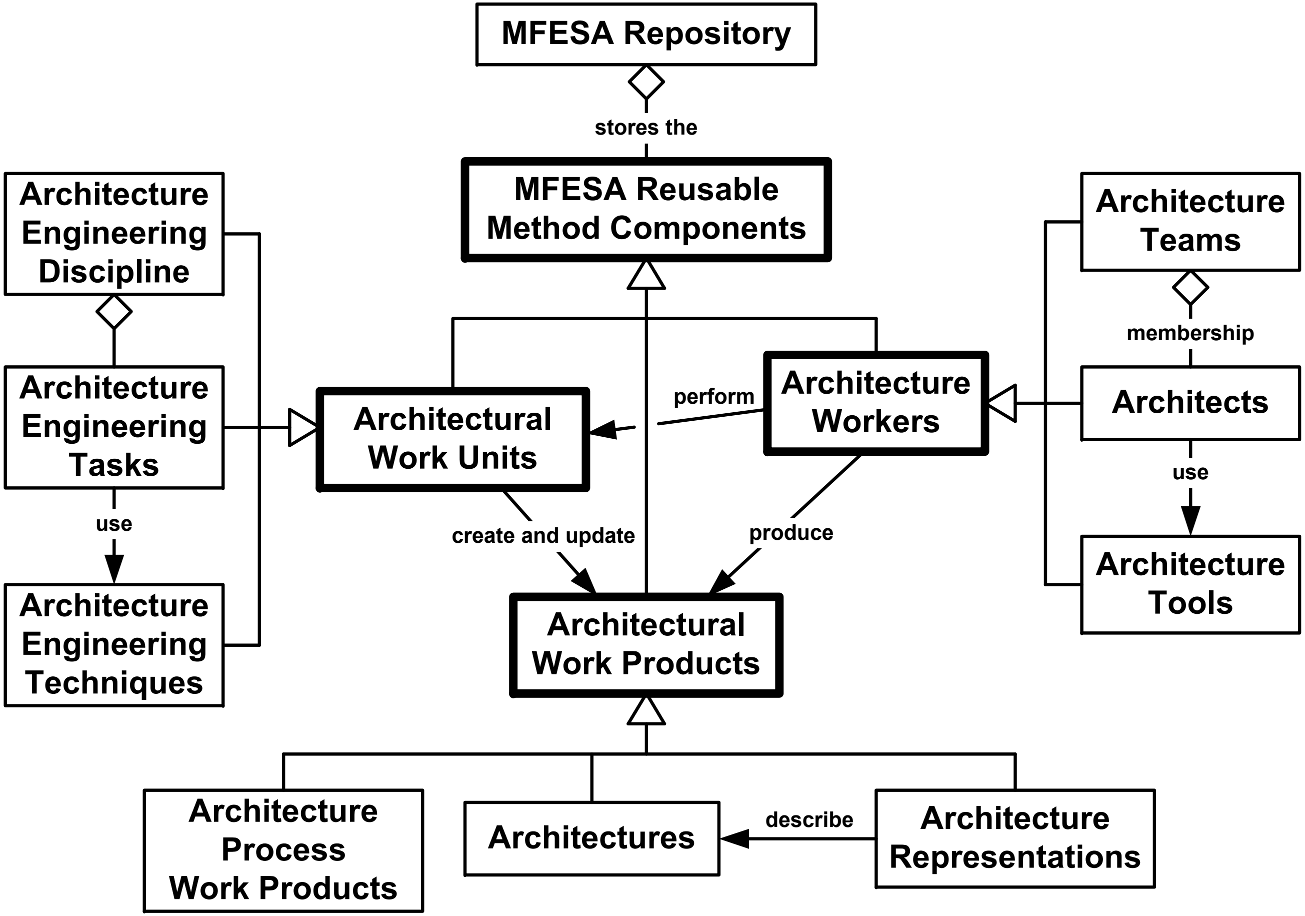
Method Engineering Models



Method vs. Process


System Architecture Engineering
documents intended way to perform
is the actual performance of
System Architecture Engineering Method Components
System Architecture Engineering Method
documents the intended
System Architecture Engineering Process
documents concrete subtypes of
consists of instances of
Architectural Workers
perform
produce
Architectural Work Units
create and modify
Architectural Work Products

MFESA Metamodel of Reusable Method Components


MFESA Repository
stores the
MFESA Reusable Method Components
Architecture Engineering Discipline
Architecture Engineering Tasks
use
Architecture Engineering Techniques
Architectural Work Units
perform
Architecture Workers
Architecture Teams
membership
Architects
use
Architecture Tools
create and update
produce
Architectural Work Products
Architecture Process Work Products
Architectures
describe
Architecture Representations

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

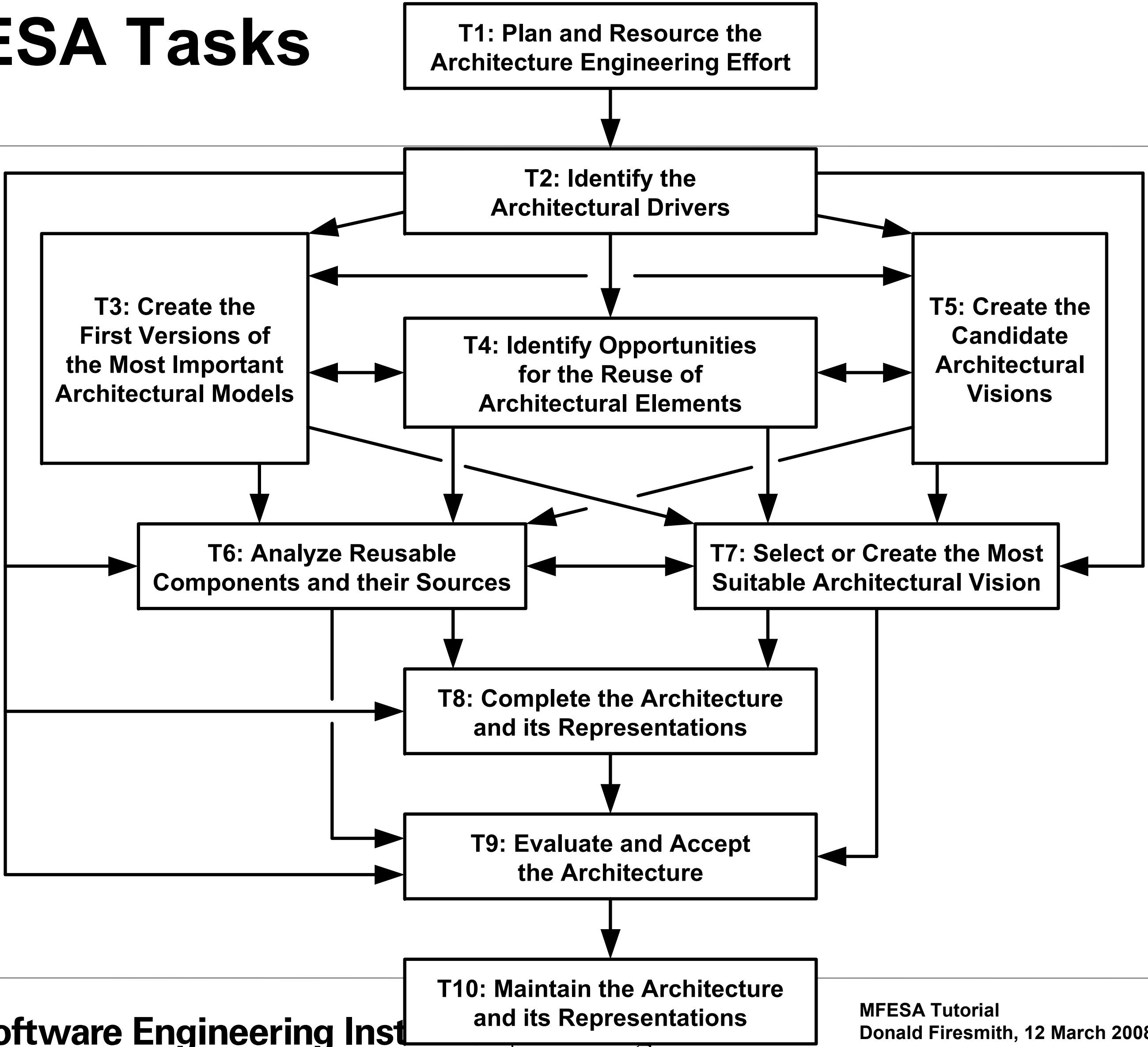
MFESA Repository of Reusable Method Components

- Architectural Work Units and Work Products ◄
- Architectural Workers

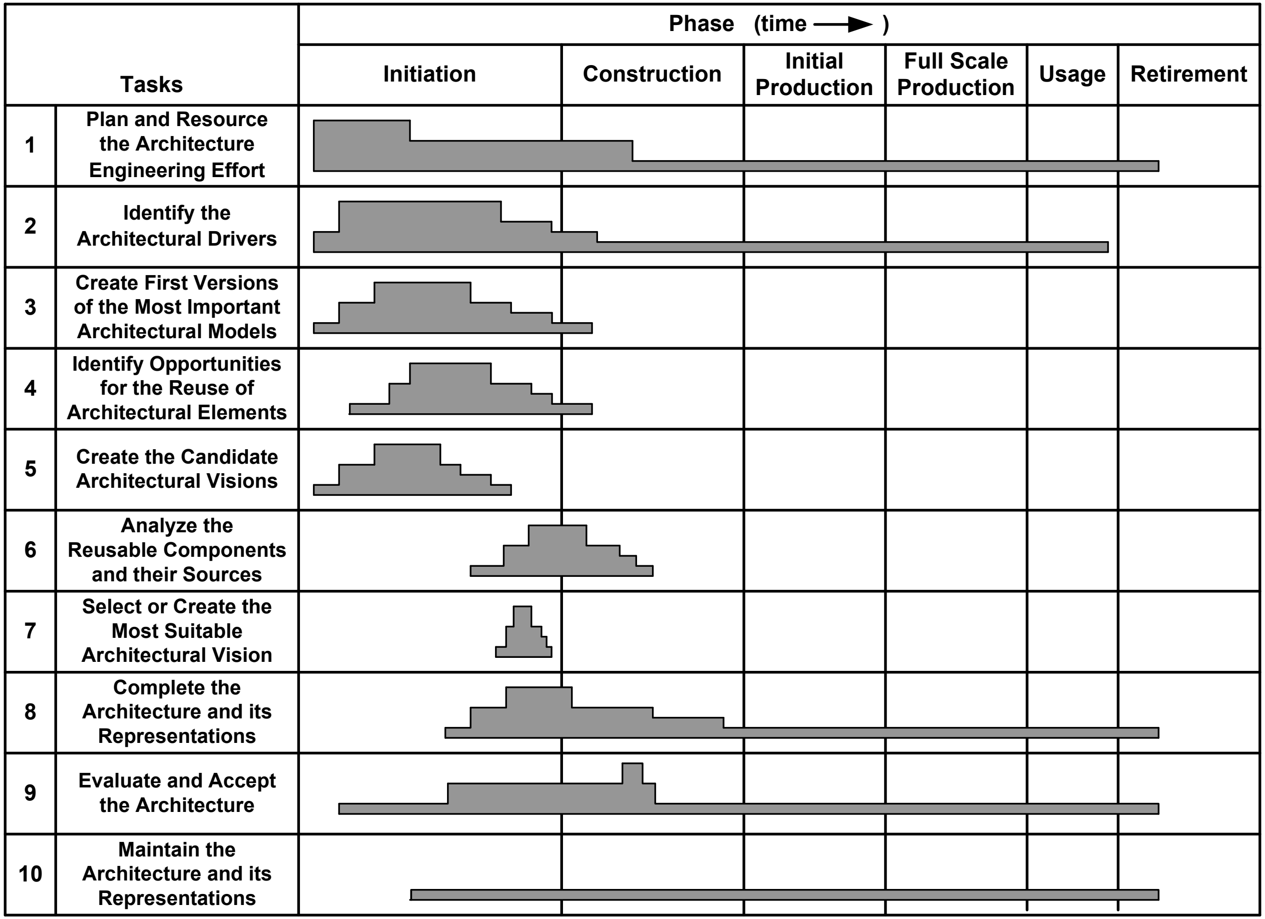
MFESA Metamethod

Conclusion

MFESA Tasks

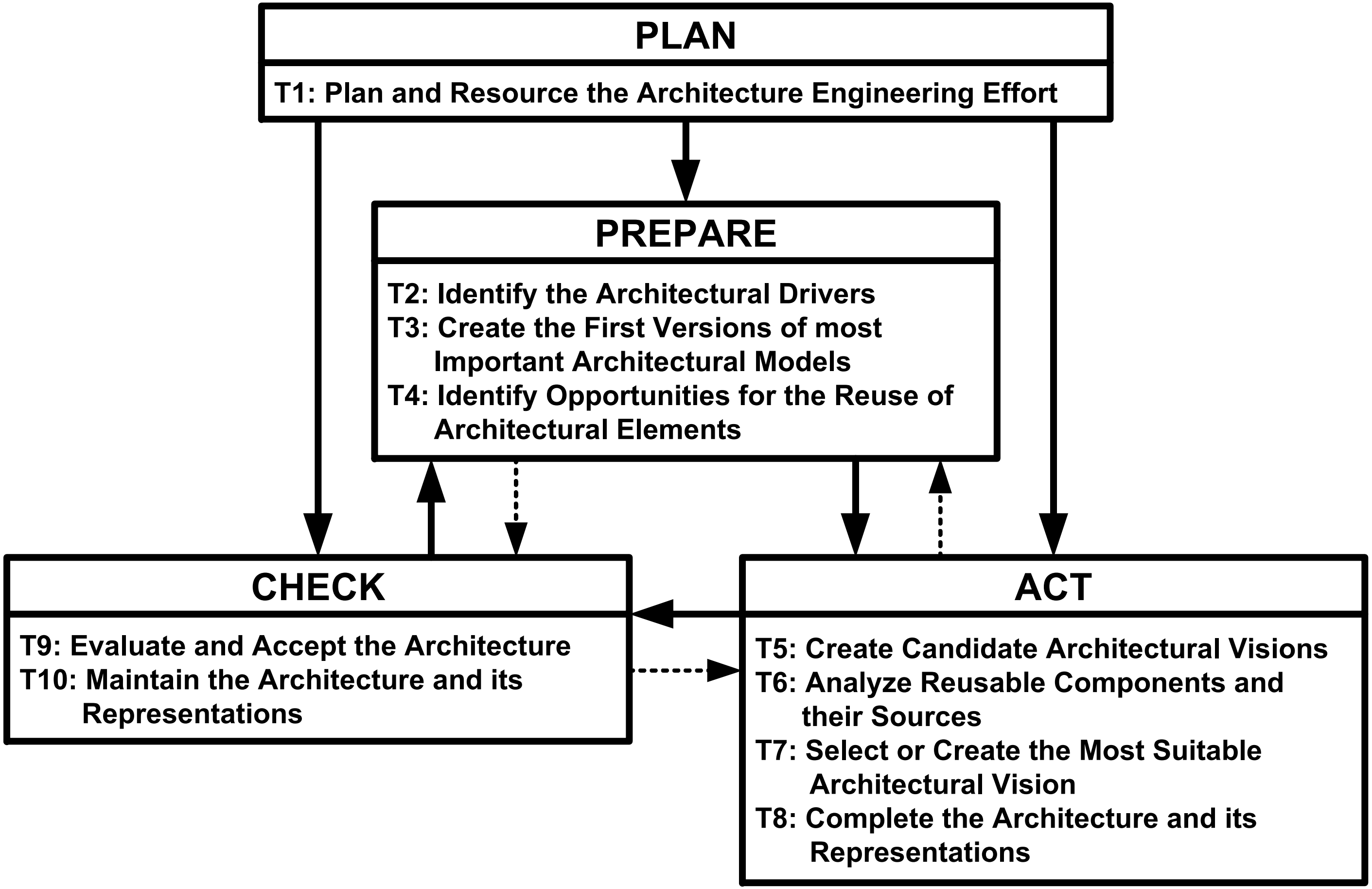

T1: Plan and Resource the Architecture Engineering Effort
T2: Identify the Architectural Drivers
T3: Create the First Versions of the Most Important Architectural Models
T4: Identify Opportunities for the Reuse of Architectural Elements
T5: Create the Candidate Architectural Visions
T6: Analyze Reusable Components and their Sources
T7: Select or Create the Most Suitable Architectural Vision
T8: Complete the Architecture and its Representations
T9: Evaluate and Accept the Architecture
T10: Maintain the Architecture and its Representations

Effort by MFESA Task



	Tasks	Initiation	Construction	Initial Production	Full Scale Production	Usage	Retirement
		Phase (time ➞)					
1	Plan and Resource the Architecture Engineering Effort						
2	Identify the Architectural Drivers						
3	Create First Versions of the Most Important Architectural Models						
4	Identify Opportunities for the Reuse of Architectural Elements						
5	Create the Candidate Architectural Visions						
6	Analyze the Reusable Components and their Sources						
7	Select or Create the Most Suitable Architectural Vision						
8	Complete the Architecture and its Representations						
9	Evaluate and Accept the Architecture						
10	Maintain the Architecture and its Representations						

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

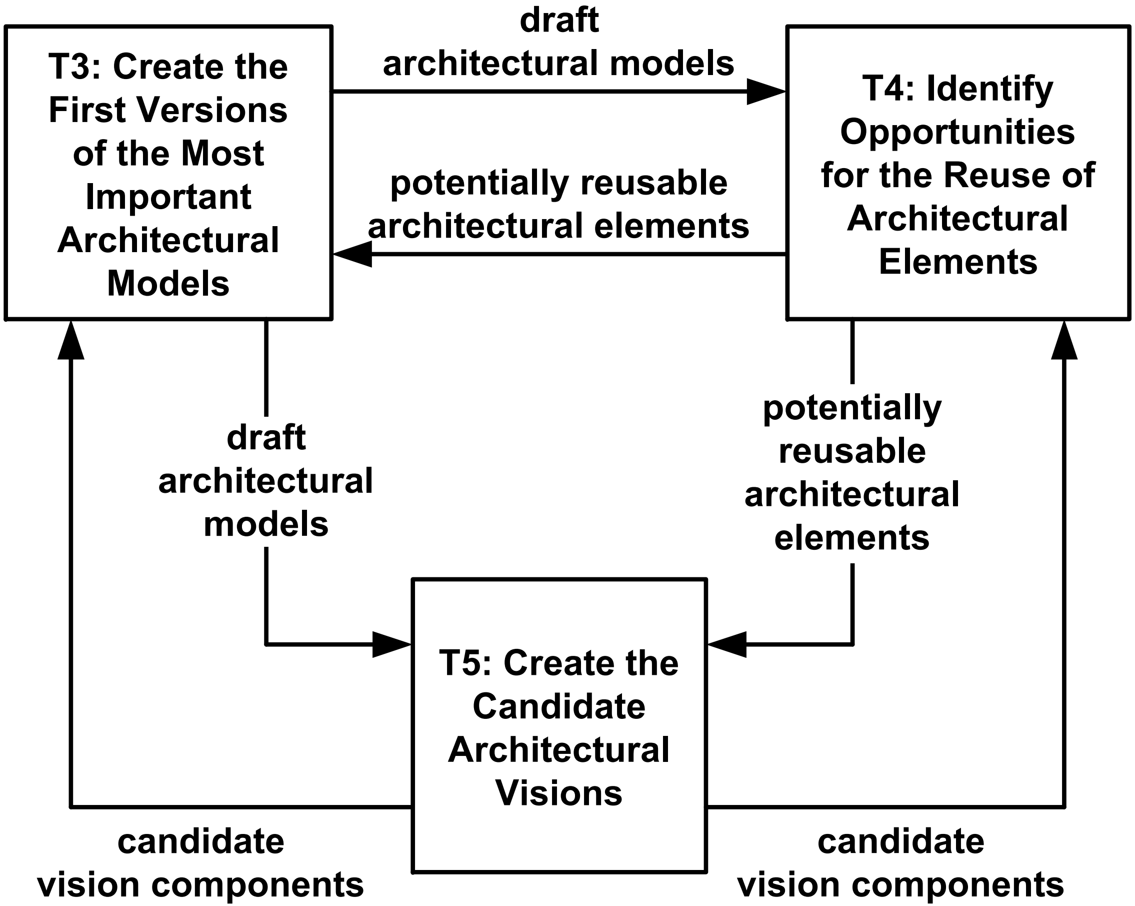
CHECK

T9: Evaluate and Accept the Architecture
T10: Maintain the Architecture and its Representations

ACT

T5: Create Candidate Architectural Visions
T6: Analyze Reusable Components and their Sources
T7: Select or Create the Most Suitable Architectural Vision
T8: Complete the Architecture and its Representations

Concurrent MFESA Tasks



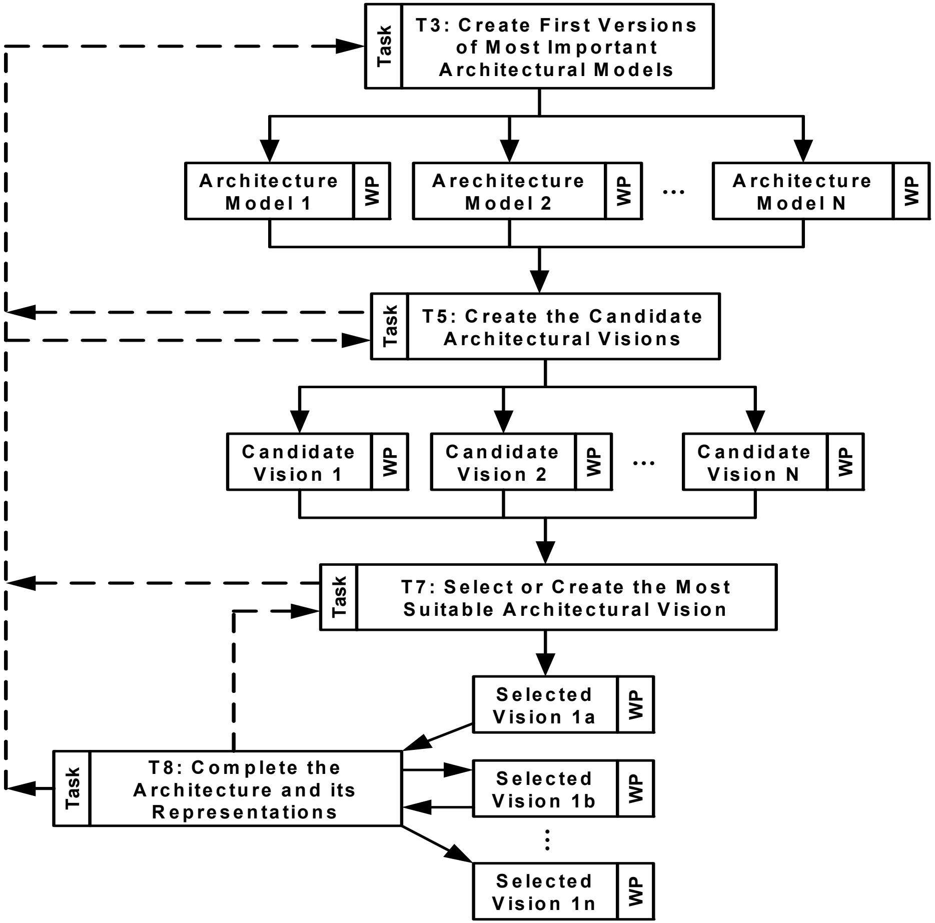
T3: Create the First Versions of the Most Important Architectural Models

T4: Identify Opportunities for the Reuse of Architectural Elements

T5: Create the Candidate Architectural Visions

draft architectural models
potentially reusable architectural elements
draft architectural models
potentially reusable architectural elements
candidate vision components
candidate vision components

Architectural Visions - Flow


Task
T3: Create First Versions of Most Important Architectural Models
Architecture Model 1
WP
Arechitecture Model 2
WP
…
Architecture Model N
WP
Task
T5: Create the Candidate Architectural Visions
Candidate Vision 1
WP
Candidate Vision 2
WP
…
Candidate Vision N
WP
Task
T7: Select or Create the Most Suitable Architectural Vision
Selected Vision 1a
WP
Task
T8: Complete the Architecture and its Representations
Selected Vision 1b
WP
⋮
Selected Vision 1n
WP

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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 1) Plan and Resource the Architecture Engineering Effort

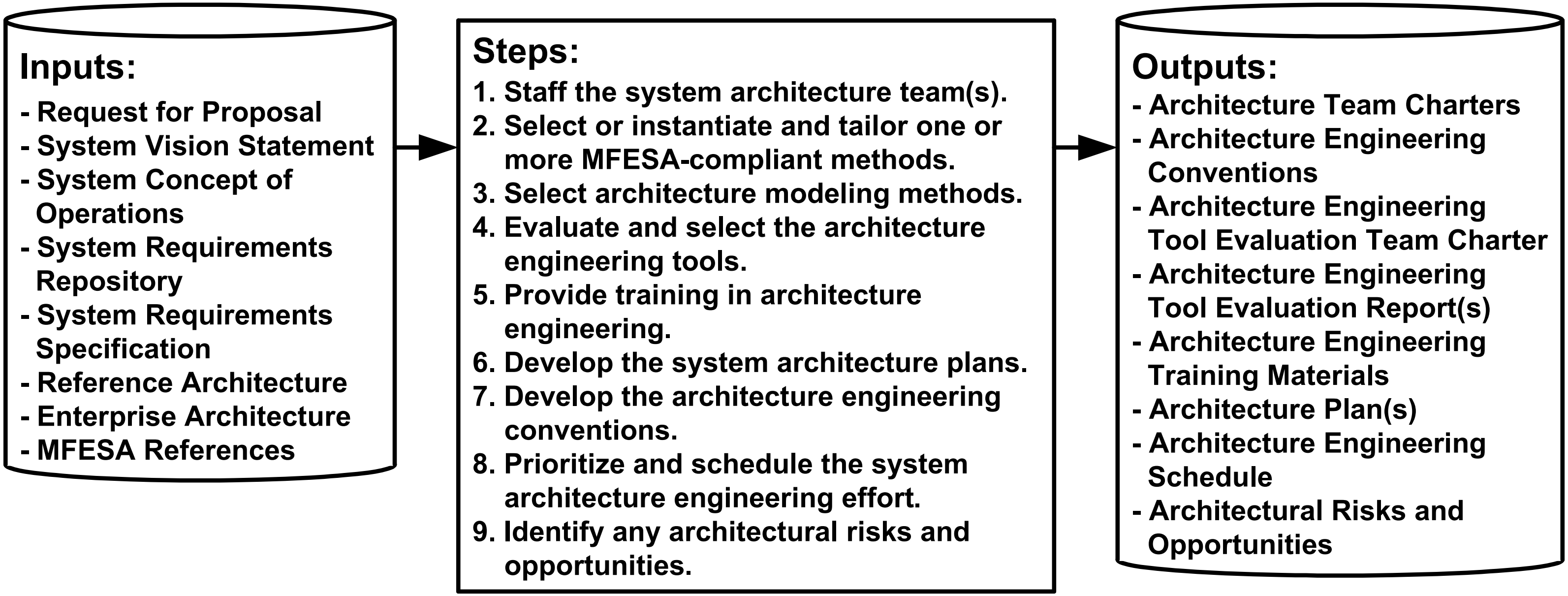
Goal:

- 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
- System Vision Statement
- System Concept of Operations
- System Requirements Repository
- 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 plans.
7. Develop the architecture engineering conventions.
8. Prioritize and schedule the system architecture engineering effort.
9. Identify any architectural risks and opportunities.

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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 2)
Identify the Architectural Drivers

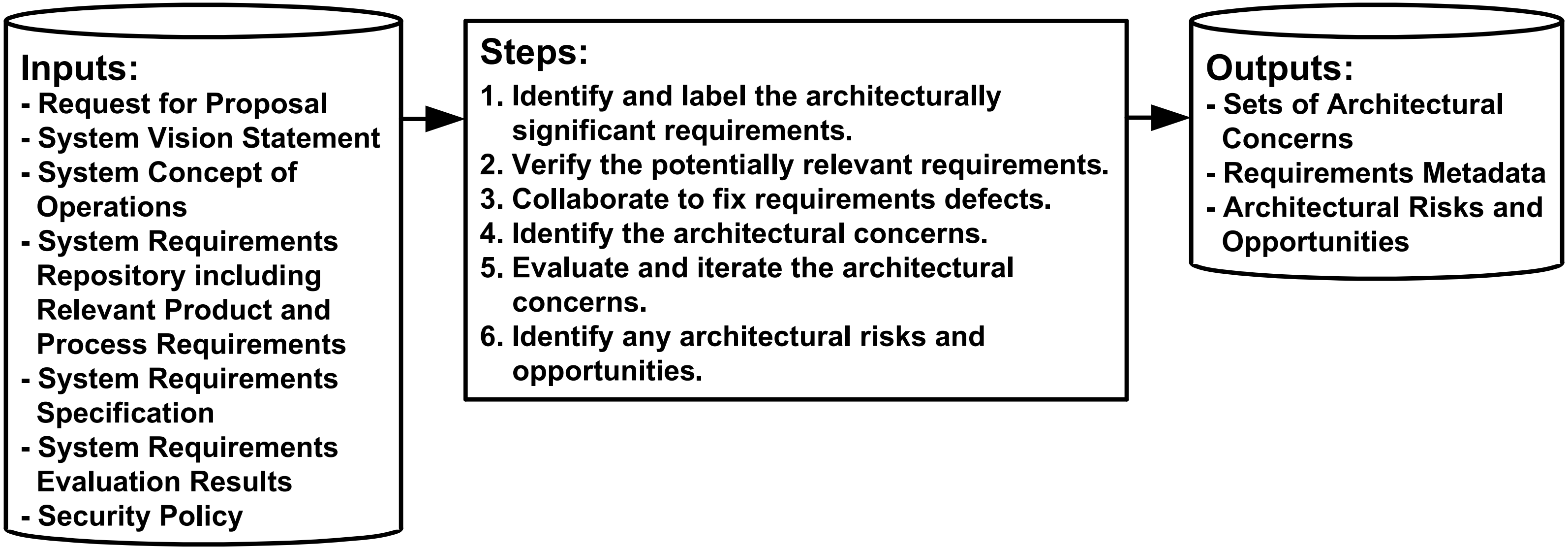
Goal:

- 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
- System Requirements Repository including Relevant Product and Process Requirements
- System Requirements Specification
- System Requirements Evaluation Results
- Security Policy

Steps:

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

Outputs:

- Sets of Architectural Concerns
- Requirements Metadata
- 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.
- 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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 3) Create Initial Architectural Models

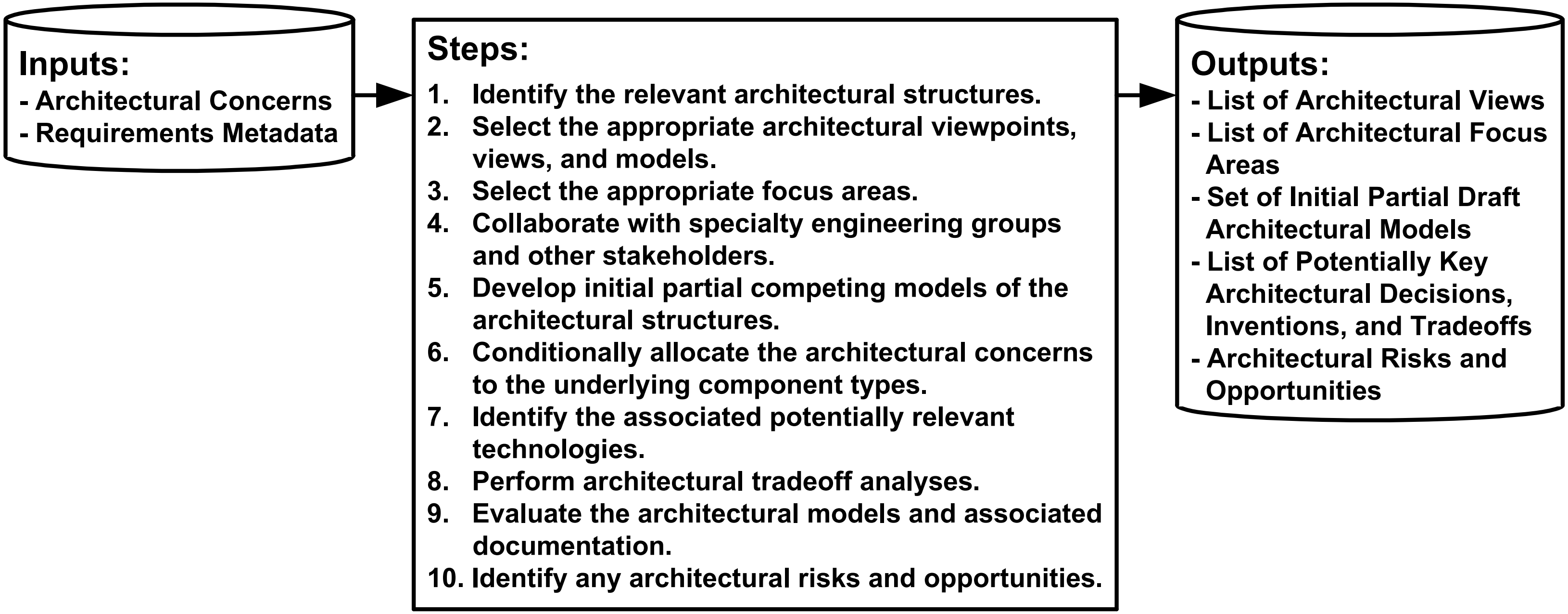
Goal:

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

Objectives:

- Capture the most important candidate system architectural decisions.
- Provide the most important views and focus areas of the system architecture.
- Ensure that the candidate architectural decisions 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
- Requirements Metadata

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 architectural risks and opportunities.

Outputs:
- List of Architectural Views
- List of Architectural Focus Areas
- Set of Initial Partial Draft Architectural Models
- List of Potentially Key Architectural Decisions, Inventions, and Tradeoffs
- 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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements

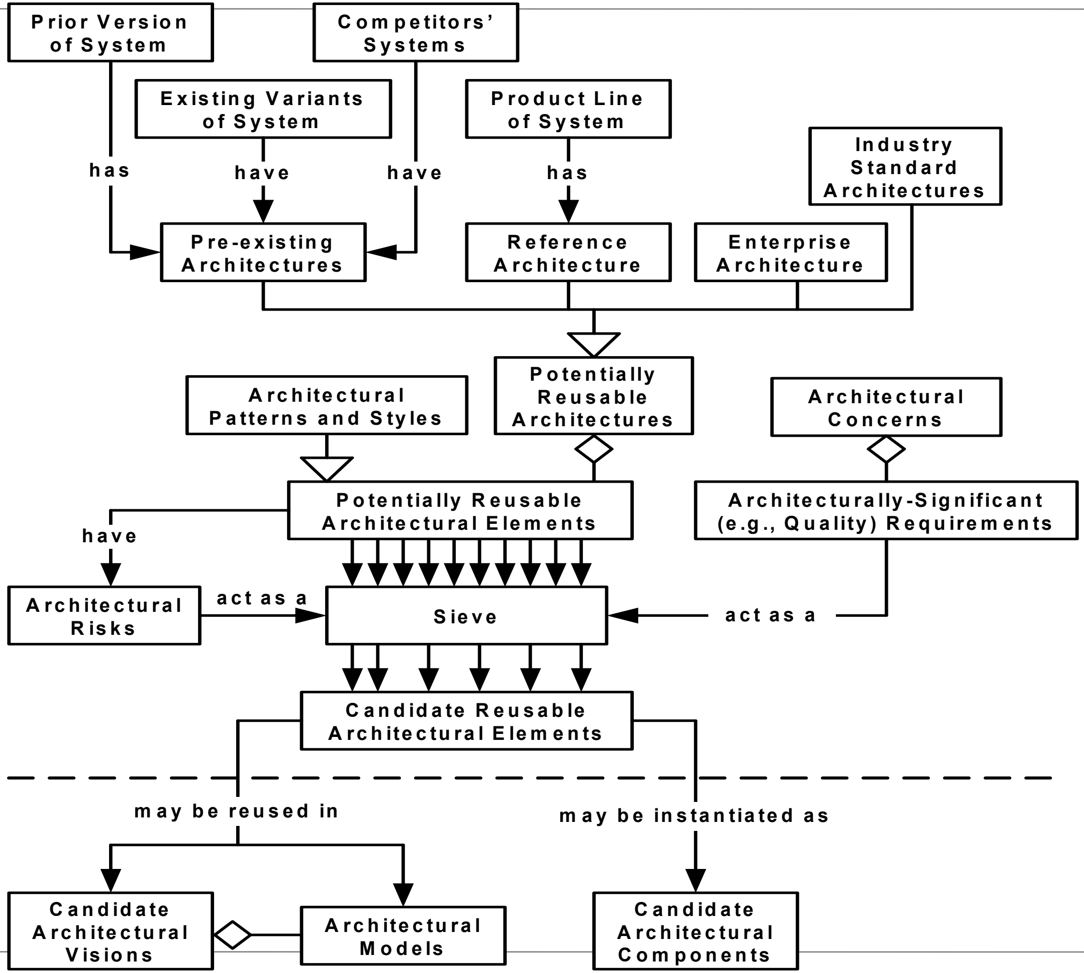
Goal:

- 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.

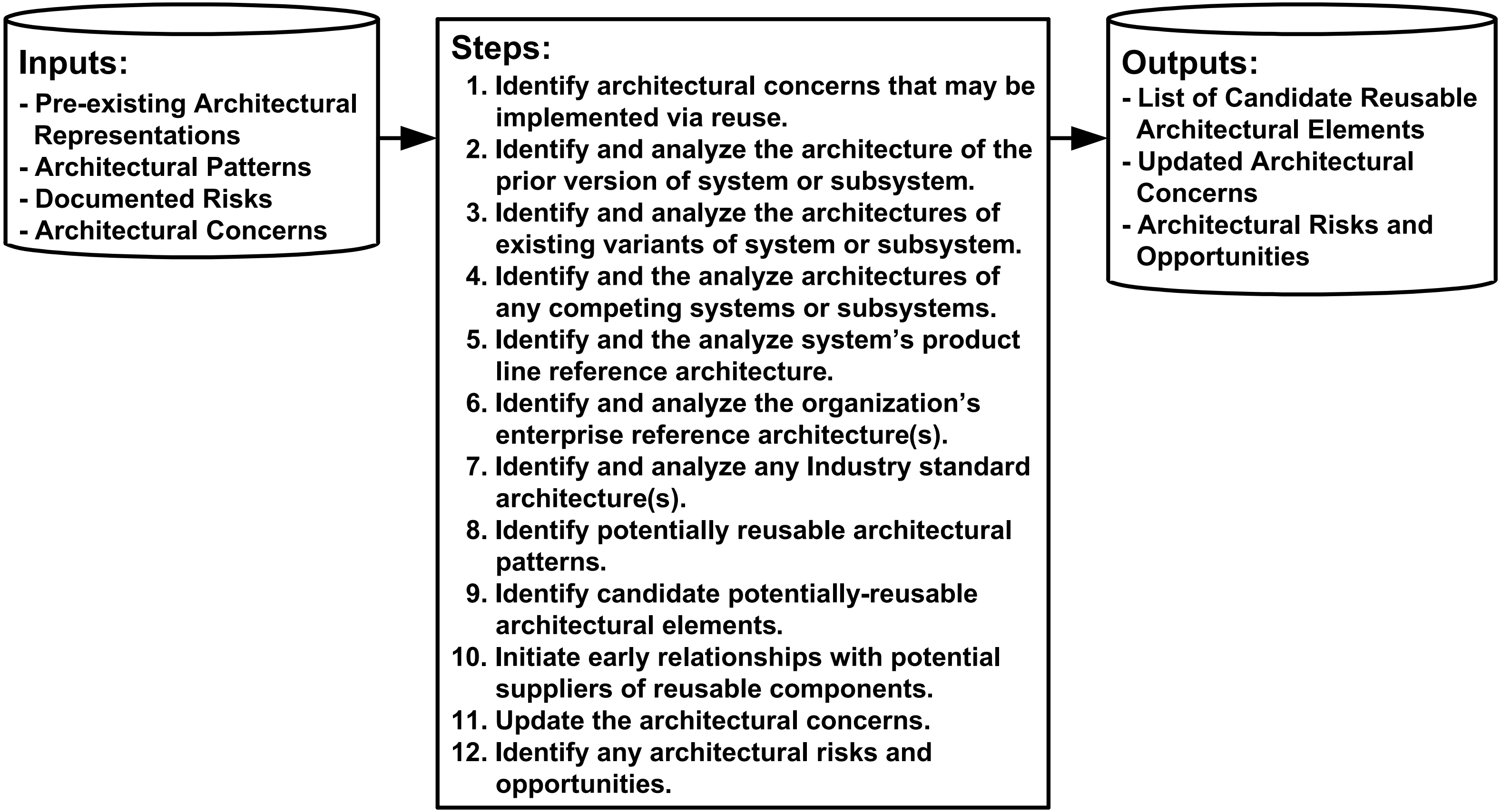
Objectives:

- 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


Prior Version of System
Competitors' Systems
Existing Variants of System
Product Line of System
Industry Standard Architectures
has
have
have
has
Pre-existing Architectures
Reference Architecture
Enterprise Architecture
Potentially Reusable Architectures
Architectural Patterns and Styles
Architectural Concerns
Potentially Reusable Architectural Elements
Architecturally-Significant (e.g., Quality) Requirements
have
Architectural Risks
act as a
Sieve
act as a
Candidate Reusable Architectural Elements
may be reused in
may be instantiated as
Candidate Architectural Visions
Architectural Models
Candidate Architectural Components

MFESA Task 4) Identify Opportunities for Reuse of Architectural Elements



Inputs:

- Pre-existing Architectural Representations
- Architectural Patterns
- Documented Risks
- Architectural Concerns

Steps:

1. Identify architectural concerns that may be implemented via reuse.
2. Identify and analyze the architecture of the prior version of system or subsystem.
3. Identify and analyze the architectures of existing variants of system or subsystem.
4. Identify and the analyze architectures of any competing systems or subsystems.
5. Identify and the analyze 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 architectural risks and opportunities.

Outputs:

- List of Candidate Reusable Architectural Elements
- Updated Architectural Concerns
- 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.
- Support modeling.
- 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 architectural components that are not ready.
- 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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 5) Create Candidate Architectural Visions

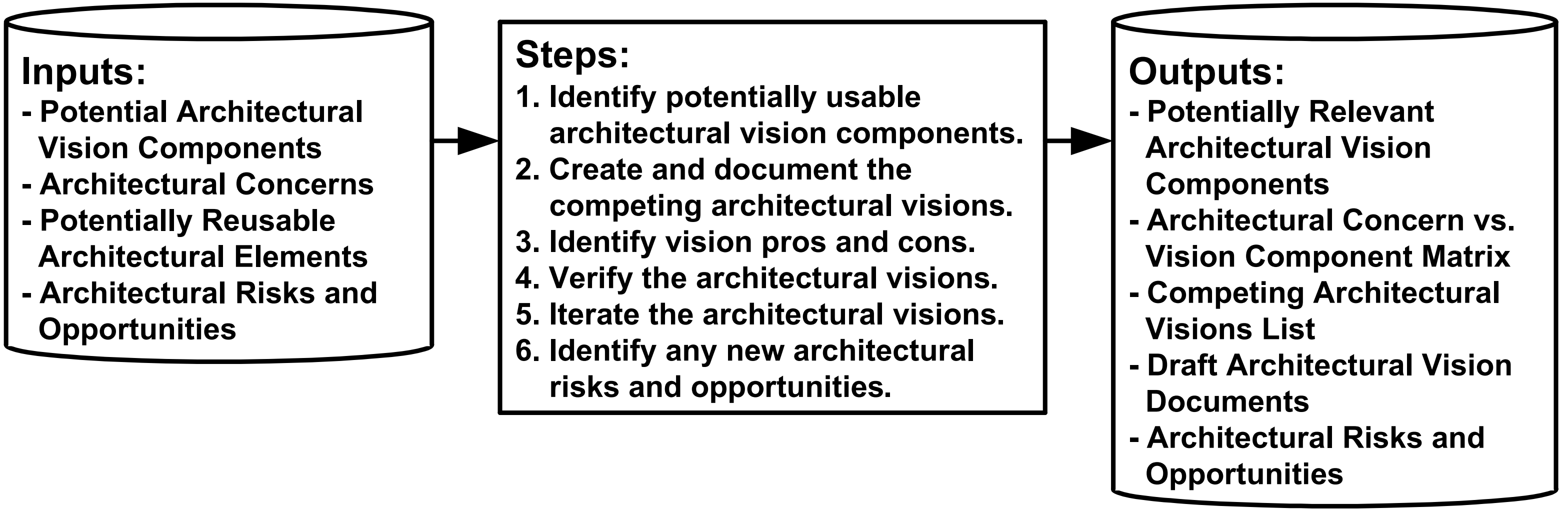
Goal:

- 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:

- Potential Architectural Vision Components
- Architectural Concerns
- Potentially Reusable Architectural Elements
- 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:

- Potentially Relevant Architectural Vision Components
- Architectural Concern vs. Vision Component Matrix
- Competing Architectural Visions List
- Draft Architectural Vision Documents
- Architectural Risks and Opportunities

MFESA Task 5)
Create Candidate Architectural Visions

Architectural Vision Component vs. Architectural Vision Matrix		Candidate Architectural Visions			
		Architectural Vision 1	Architectural Vision 2	Architectural Vision 3	Architectural Vision 4
Candidate Architectural Vision Components	Component 1	X	X	X	X
	Component 2	X		X	X
	Component 3	X	X		
	Component 4	X	X	X	
	Component 5	X		X	X
	Component 6	X	X	X	
	Component 7	X		X	X
	Component 8	X	X	X	X
	Component 9		X		X
	Component 10		X	X	X
	Component 11			X	
	Component 12			X	X
	Component 13			X	X

MFESA Task 5)
Create Candidate Architectural Visions

Example Architectural Concern vs. Vision Component Matrix

Architectural Concern vs. Vision Component Matrix		Architectural Vision Components												
		User Identifier and Pass Phrase	COTS Security Smart Card	COTS Biometrics Reader	Locked Rooms with Keyed Access	Guards with Security Cameras	Dedicated Encryption Decryption HW Server	COTS Encryption Decryption Software	Encrypted Messages	Encrypted Database Records	COTS Intrusion Detection Software	COTS Antivirus Software	Digital Signature	Single Sign-On
Architectural Concerns	Access Control	++	++	++	++	++	0	0	0	0	0	0	+	++
	Confidentiality	+	+	+	+	+	++	++	++	++	0	0	0	0
	Integrity (Message)	+	+	+	+	+	0	0	+	+	+	++	+	0
	Integrity (Software)	+	+	+	+	+	0	0	0	0	+	++	+	0
	Integrity (Data)	+	+	+	+	+	0	0	0	+	+	++	+	0
	Nonrepudiation	0	0	0	0	0	0	0	0	+	+	+	++	0
	Availability	-	-	0	0	0	0	0	0	-	+	+	0	0
	Cost	++	-	- -	0	0	- -	- -	- -	0	-	+	-	-
	Performance	-	-	0	0	0	+	-	-	- -	-	-	-	+
	Usability	-	-	0	0	0	0	0	0	0	0	0	-	++

MFESA Task 5) Create Candidate Architectural Visions

Guidelines

- Identify an appropriate number of candidate architectural visions.
- Complete candidate architectural visions to appropriate level of detail.
- Prepare architectural components for OTS incorporation.
- 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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 6) Analyze Reusable Components and their Sources

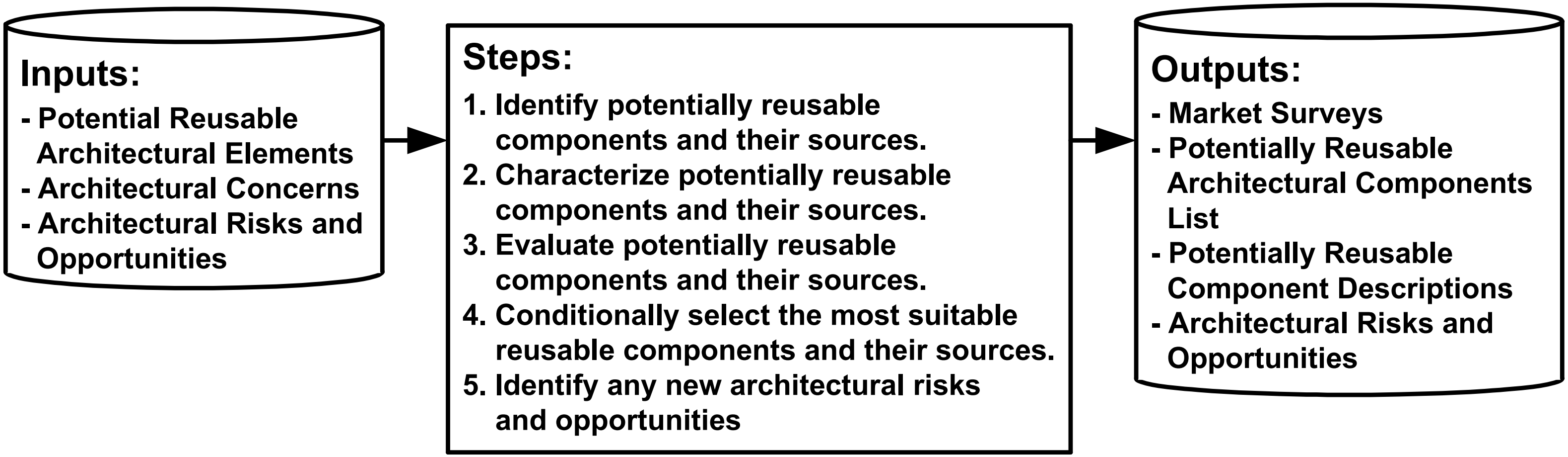
Goal:

- 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:

- Potential Reusable Architectural Elements
- Architectural Concerns
- Architectural Risks and Opportunities

Steps:

1. Identify potentially reusable components and their sources.
2. Characterize potentially reusable components and their sources.
3. Evaluate 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
- Architectural Risks and Opportunities

MFESA Task 6) Analyze Reusable Components and their Sources

Guidelines

- Use appropriate decision techniques.
- Perform task 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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 7) Select or Create the Most Suitable Architectural Vision

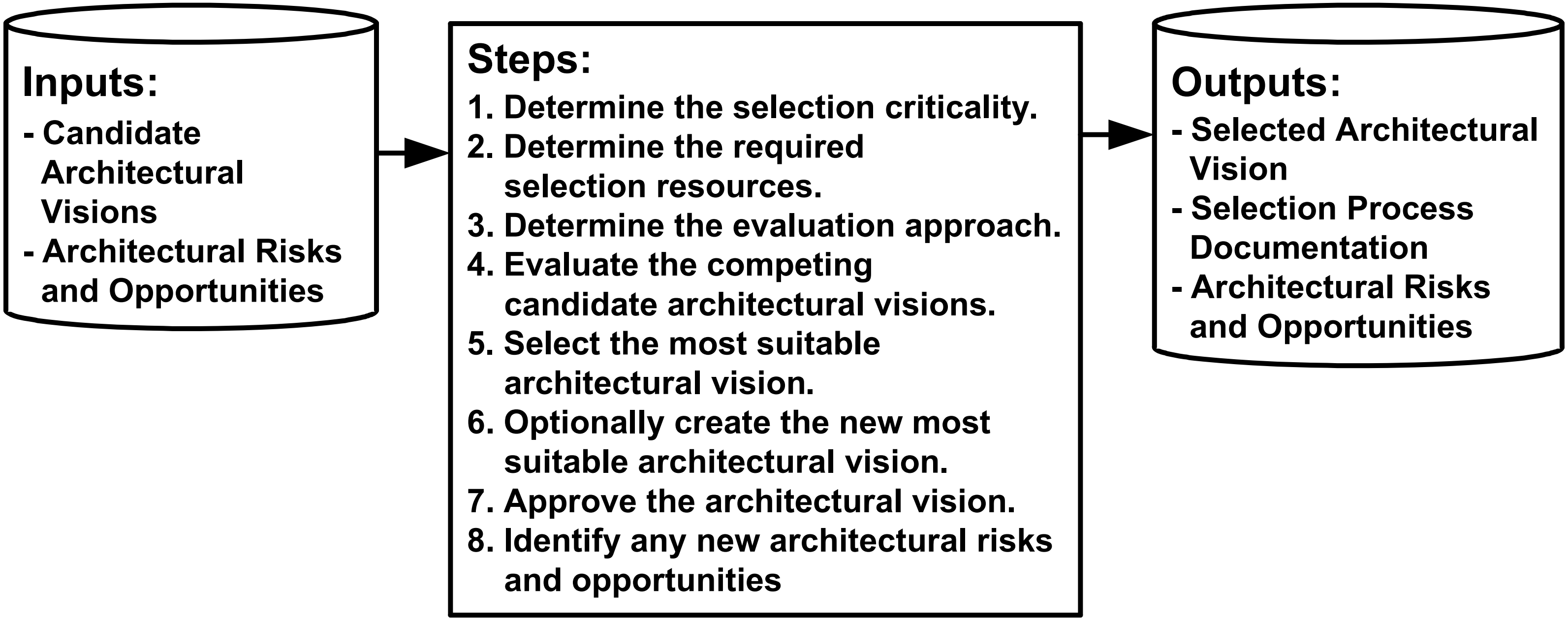
Goal:

- 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:

- Candidate Architectural Visions
- 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:

- Selected Architectural Vision
- Selection Process Documentation
- Architectural Risks and Opportunities

MFESA Task 7) Select or Create the Most Suitable Architectural Vision

Architectural Concern vs. Architectural Visions Matrix		Candidate Competing Architectural Visions			
		Architectural Vision 1	Architectural Vision 2	Architectural Vision 3	Architectural Vision 4
Architectural Concerns	Availability	+	+	++	+
	Development Cost	0	++	--	0
	Development Schedule	+	++	--	-
	Interoperability	+	-	+	+
	Performance	+	--	+	++
	Portability	0	+	0	+
	Reliability	+	-	++	-
	Safety	-	-	++	0
	Security	-	++	-	0
	Usability	-	0	-	+

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 the Architecture and its Representations

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 the Architecture and its Representations ◄

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations

MFESA Task 8) Complete the Architecture and its Representations

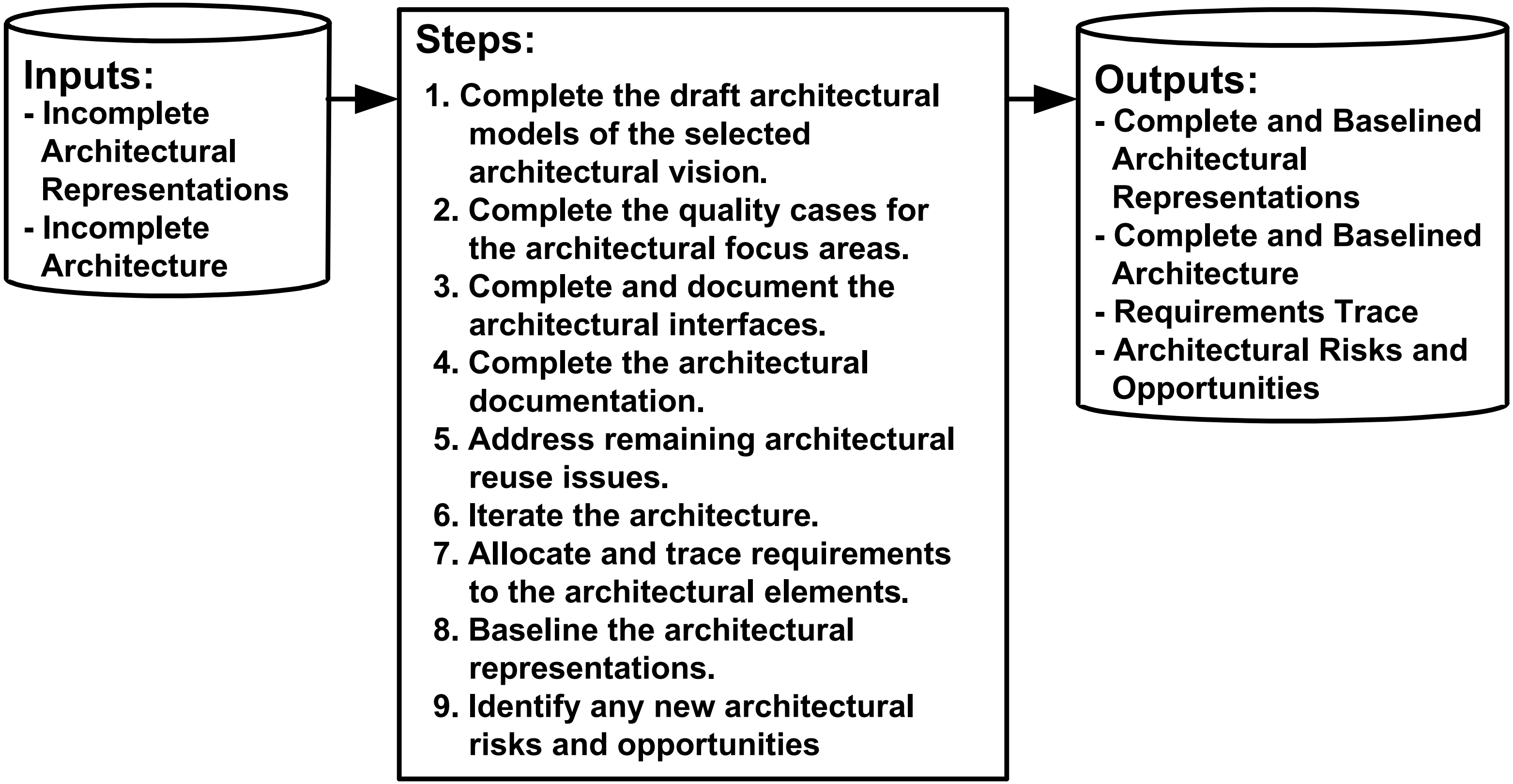
Goal:

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

Objectives:

- Complete the architectural models.
- Complete the interface aspects of the architectural.
- Complete the reuse aspects of the architecture.
- Complete the architectural representations.
- Provide a system or subsystem architecture that can be evaluated and accepted by its authoritative stakeholders .

MFESA Task 8) Complete the Architecture and its Representations



Inputs:

- Incomplete Architectural Representations
- Incomplete Architecture

Steps:

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

Outputs:

- Complete and Baselined Architectural Representations
- Complete and Baselined Architecture
- Requirements Trace
- Architectural Risks and Opportunities

MFESA Task 8) Complete the Architecture and its Representations

Guidelines

- Develop quality cases as a natural part of the architecture engineering process.
- Architect all relevant types of interfaces.
- Work with the requirements team to provide requirements traceability.
- Formally manage architectural risks.

MFESA Task 8) Complete the Architecture and its Representations

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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture ◄

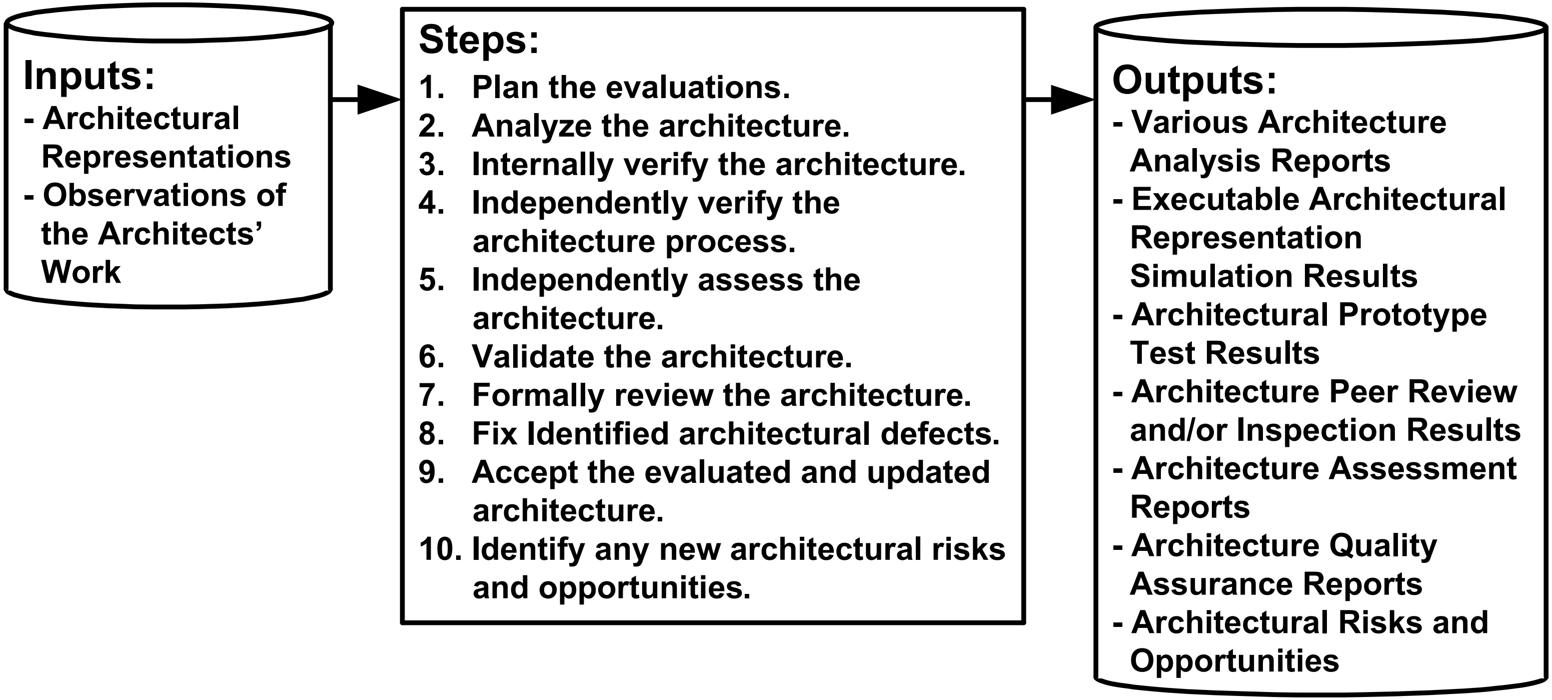
Task 10) Maintain the Architecture and its Representations

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



Inputs:

- Architectural Representations
- Observations of the Architects' Work

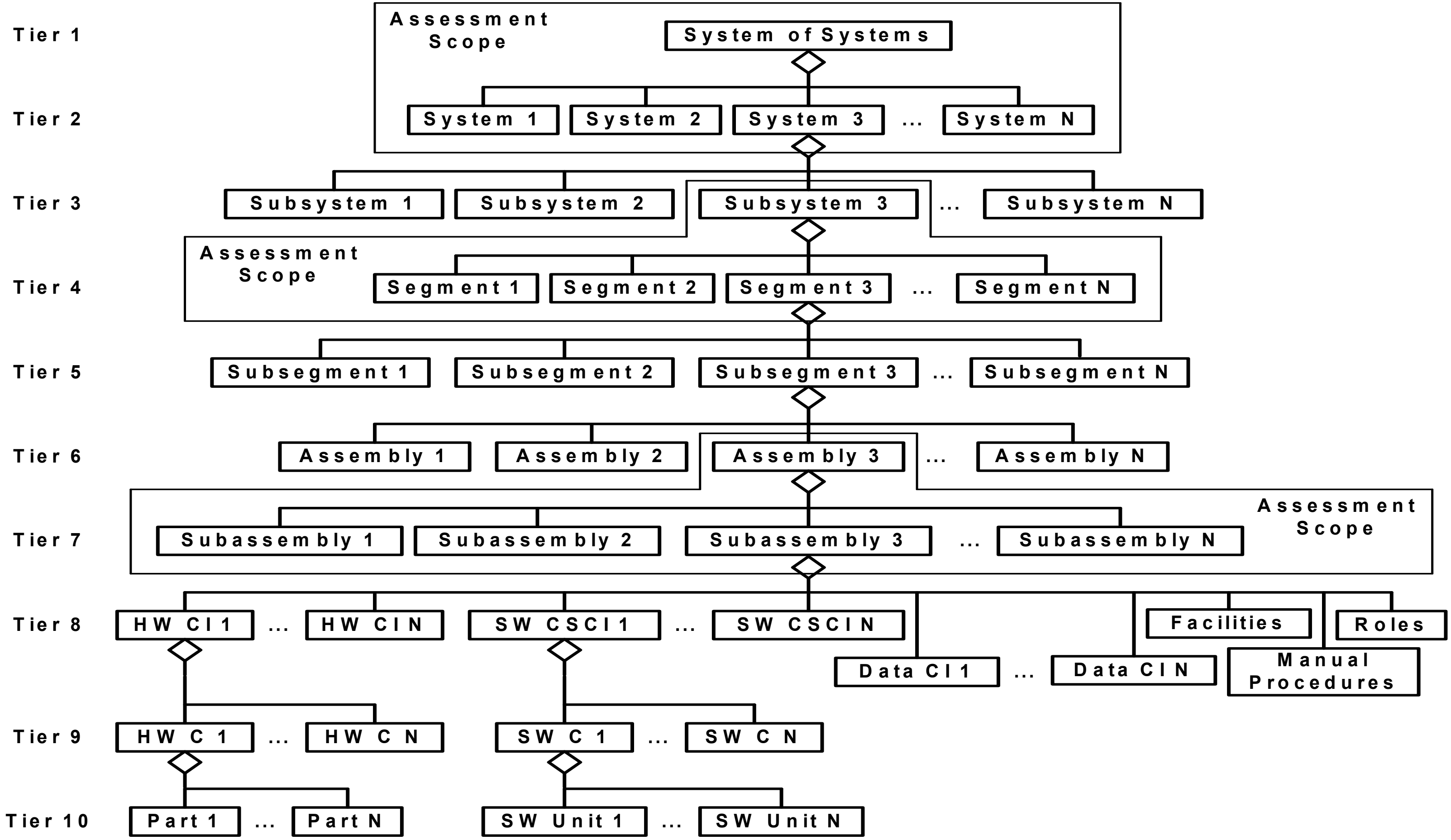
Steps:

1. Plan the evaluations.
2. Analyze the architecture.
3. Internally verify the architecture.
4. Independently verify the architecture process.
5. Independently assess the architecture.
6. Validate the architecture.
7. Formally review the architecture.
8. Fix Identified architectural defects.
9. Accept the evaluated and updated architecture.
10. Identify any new architectural risks and opportunities.

Outputs:

- Various Architecture Analysis Reports
- Executable Architectural Representation Simulation Results
- Architectural Prototype Test Results
- Architecture Peer Review and/or Inspection Results
- Architecture Assessment Reports
- Architecture Quality Assurance Reports
- Architectural Risks and Opportunities

MFESA Task 9) Evaluate and Accept the Architecture



MFESA Task 9) Evaluate and Accept the Architecture

Guidelines

- Use evaluations to support architectural milestones.
- Evaluate continuously.
- 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) Maintain the Architecture and its Representations

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 the Architecture and its Representations

Task 9) Evaluate and Accept the Architecture

Task 10) Maintain the Architecture and its Representations ◄

MFESA Task 10) Maintain the Architecture and its Representations

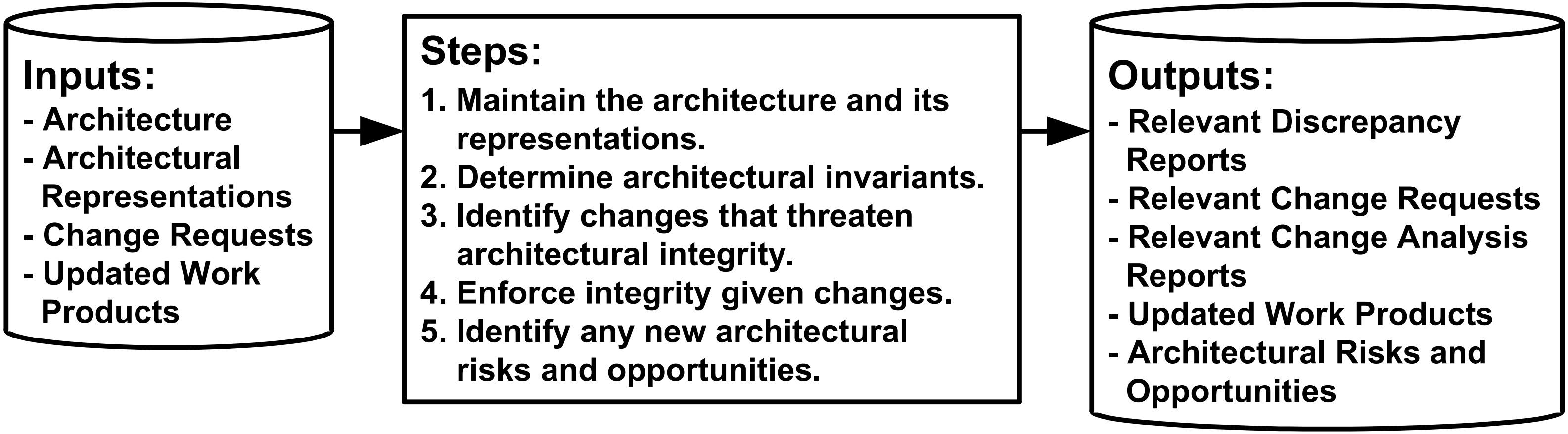
Goal:

- Maintain the system or subsystem architecture as the architecturally significant requirements change.
- 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) Maintain the Architecture and its Representations



Inputs:

- Architecture
- Architectural Representations
- Change Requests
- Updated Work Products

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 Discrepancy Reports
- Relevant Change Requests
- Relevant Change Analysis Reports
- Updated Work Products
- Architectural Risks and Opportunities

MFESA Task 10) Maintain the Architecture and its Representations

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) Maintain the Architecture and its Representations

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

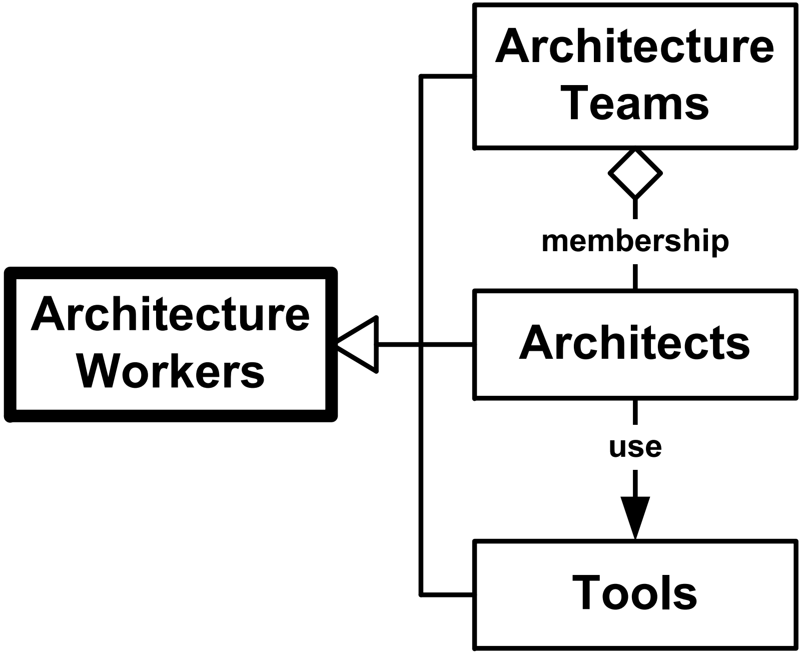
MFESA Repository of Reusable Method Components

- Architectural Work Units and Work Products
- Architectural Workers ◀

MFESA Metamethod

Conclusion

MFESA Repository – Architecture Workers

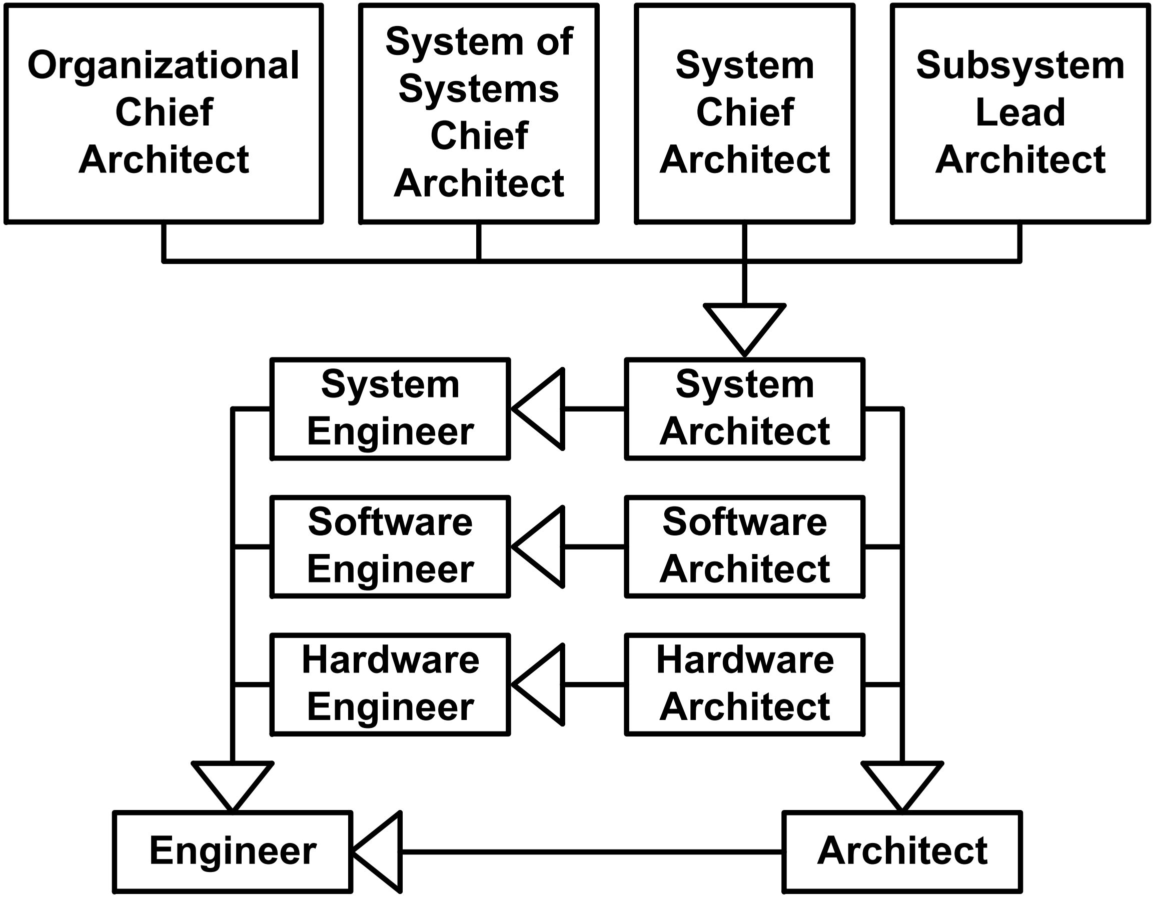


Architects - Definition

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



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

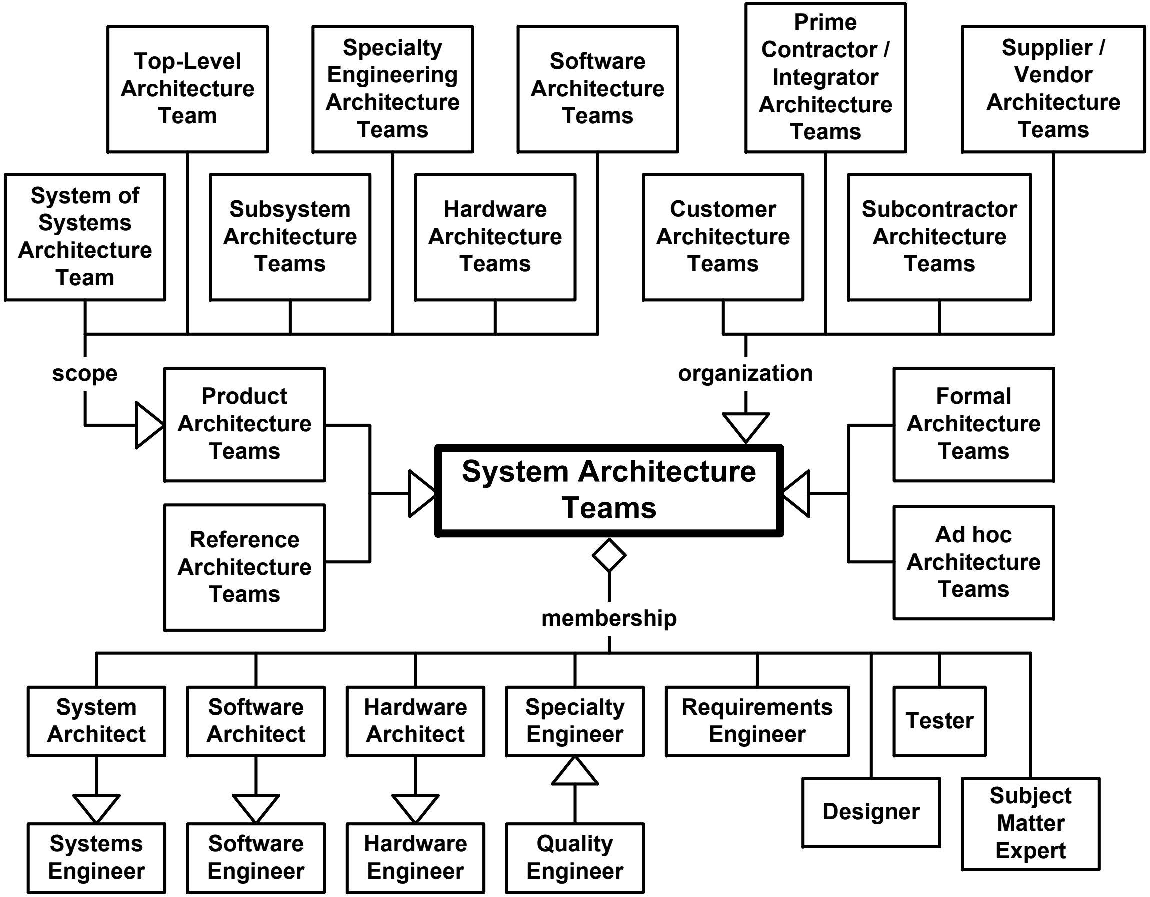
Instantiate evaluation of Off-The-Shelf architectural components

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
scope
organization
Product Architecture Teams
Reference Architecture Teams
System Architecture Teams
Formal Architecture Teams
Ad hoc Architecture Teams
membership
System Architect
Software Architect
Hardware Architect
Specialty Engineer
Requirements Engineer
Tester
Systems Engineer
Software Engineer
Hardware Engineer
Quality Engineer
Designer
Subject Matter Expert

System Architecture Tools - Definition

System Architecture Tool

any thing 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

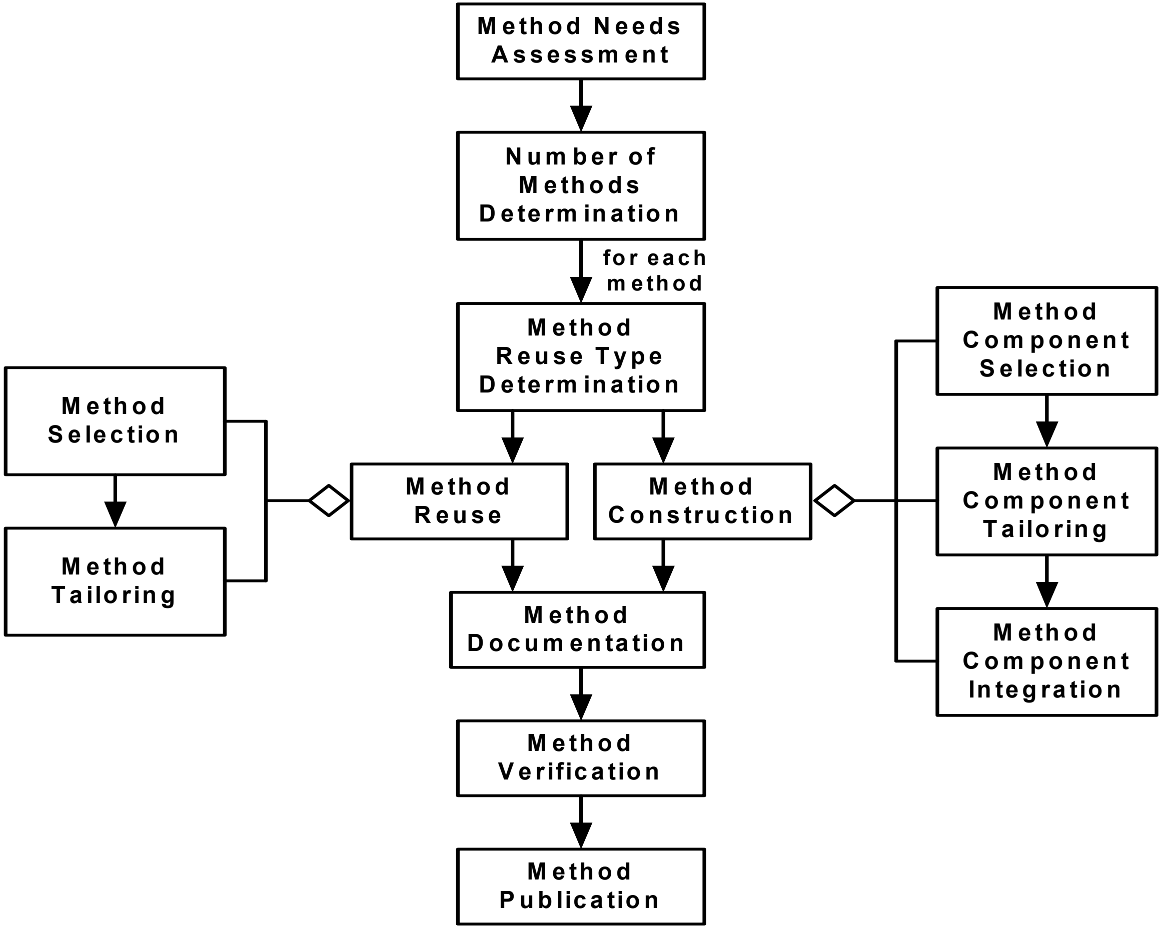
MFESA Repository of Reusable Method Components

- Architectural Work Units and Work Products
- Architectural Workers

MFESA Metamethod ◄

Conclusion

MFESA Metamethod - Tasks



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.

Future Book

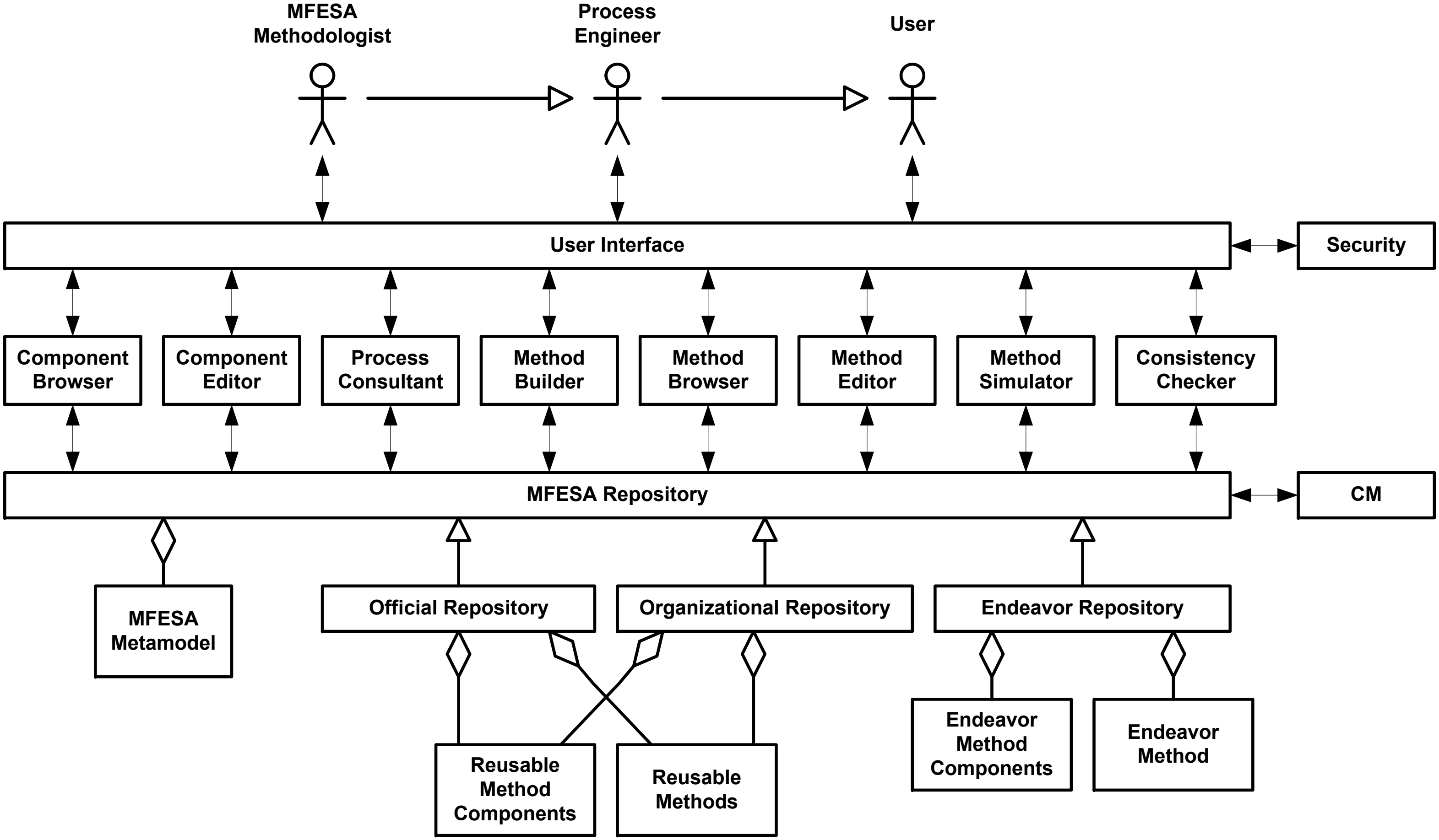
Method Framework for Engineering System Architectures (MFESA):
Generating Effective and Efficient Project-Specific
System Architecture Engineering Methods

Donald Firesmith (SEI) with Peter Capell (SEI) , Dietrich Falkenthal (MITRE), Charles Hammons (SEI), DeWitt Latimer IV (USAF), and Tom Merendino (SEI)

Auerbach Publishing

2008

Potential Future Tools



Questions?

For more information, contact:

Donald Firesmith

Acquisition Support Program

Software Engineering Institute

dgf@sei.cmu.edu