The Method Framework for Engineering System Architectures (MFESA)

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Donald Firesmith
5 March 2009

Donald G. Firesmith

A senior member of the technical staff at the SEI, Donald Firesmith works in the Acquisition Support Program (ASP) where he helps the US Department of Defense acquire large complex software-intensive systems. With over 25 years of industry experience, he has published 6 software and system engineering books in the areas of process, object orientation, and system architecture engineering. He is currently writing a book on engineering safety- and security-related requirements. He has also published dozens of technical articles, spoken at numerous international conferences, and has been the program chair or on the program committee of several conferences. He has taught several hundred courses in industry and numerous tutorials at conferences. He is also the founding chair of the OPEN Process Framework (OPF) Repository organization www.opfro.org, which provides the world's largest free open-source website documenting over 1,100 reusable method components.

Webinar Objectives

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

- **MFESA *Ontology*** of underlying architecture engineering concepts and terminology
- **MFESA *Metamodel*** of foundational types 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

Polling Questions

What is your primary job?

1. System Architect
2. Software Architect
3. System Engineer
4. Technical Leader
5. Other

What kind of an organization do you work for?

1. Government
2. Military
3. Defense Contractor
4. Commercial Contractor
5. Academia

System Architecture – A 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

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 is Critical

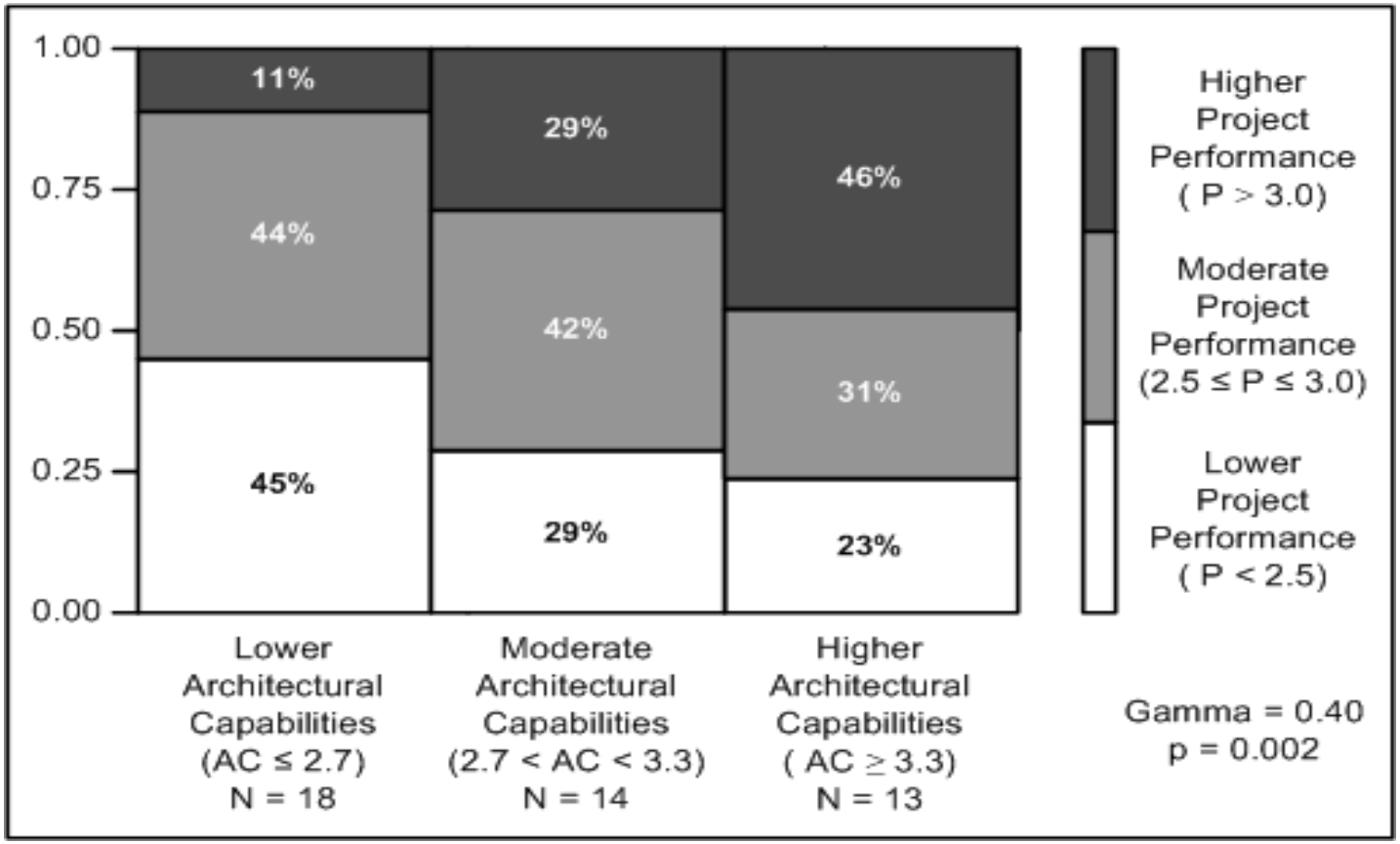
Supports achievement of critical architecturally significant requirements

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
- 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)

Why Method Engineering? – *Systems* Vary Greatly

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

Why Method Engineering? – *Systems* Vary Greatly$_2$

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 organization

Type of organizations:

- Owner, Acquirer, Developer, Operator, User, Maintainer
- Prime contractor, subcontractors, vendors, system integrator

Degree of centralized/distributed governance:

- Authority, policy, funding
- Scheduling

Management culture

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, sustainment)

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

Schedule (adequacy, criticality, coordination)

Funding (adequacy, distribution)

Why Method Engineering? – Stakeholders

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)

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.

MFESA Project

Started January 2007

Collaborators:

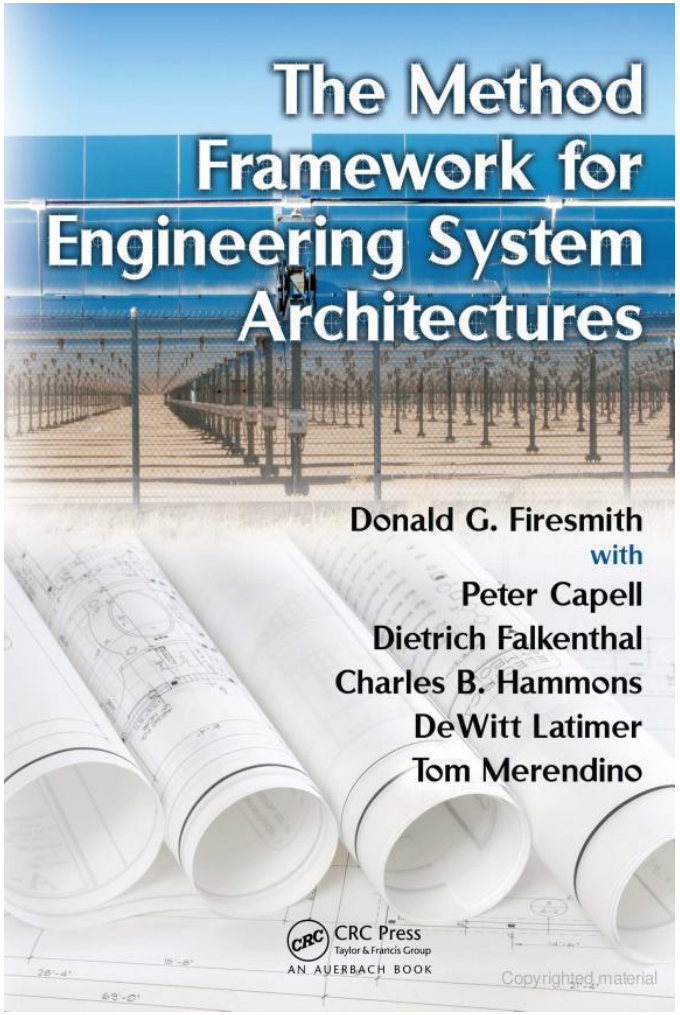
- SEI Acquisition Support Program (ASP) – Don Firesmith (Team Lead), Peter Capell, Bud Hammons, and Tom Merendino
- MITRE – Dietrich Falkenthal (Bedford MA)
- USAF – DeWitt Latimer (USC)

Current work products:

- Reference Book (CRC Press – Auerbach Publishing, November 2008)
- Tutorials and Training Materials
- Articles

Eventual work products (we hope!):

- Informational website with softcopies of the method components
- Free, open-source tool set (Eclipse ?)



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

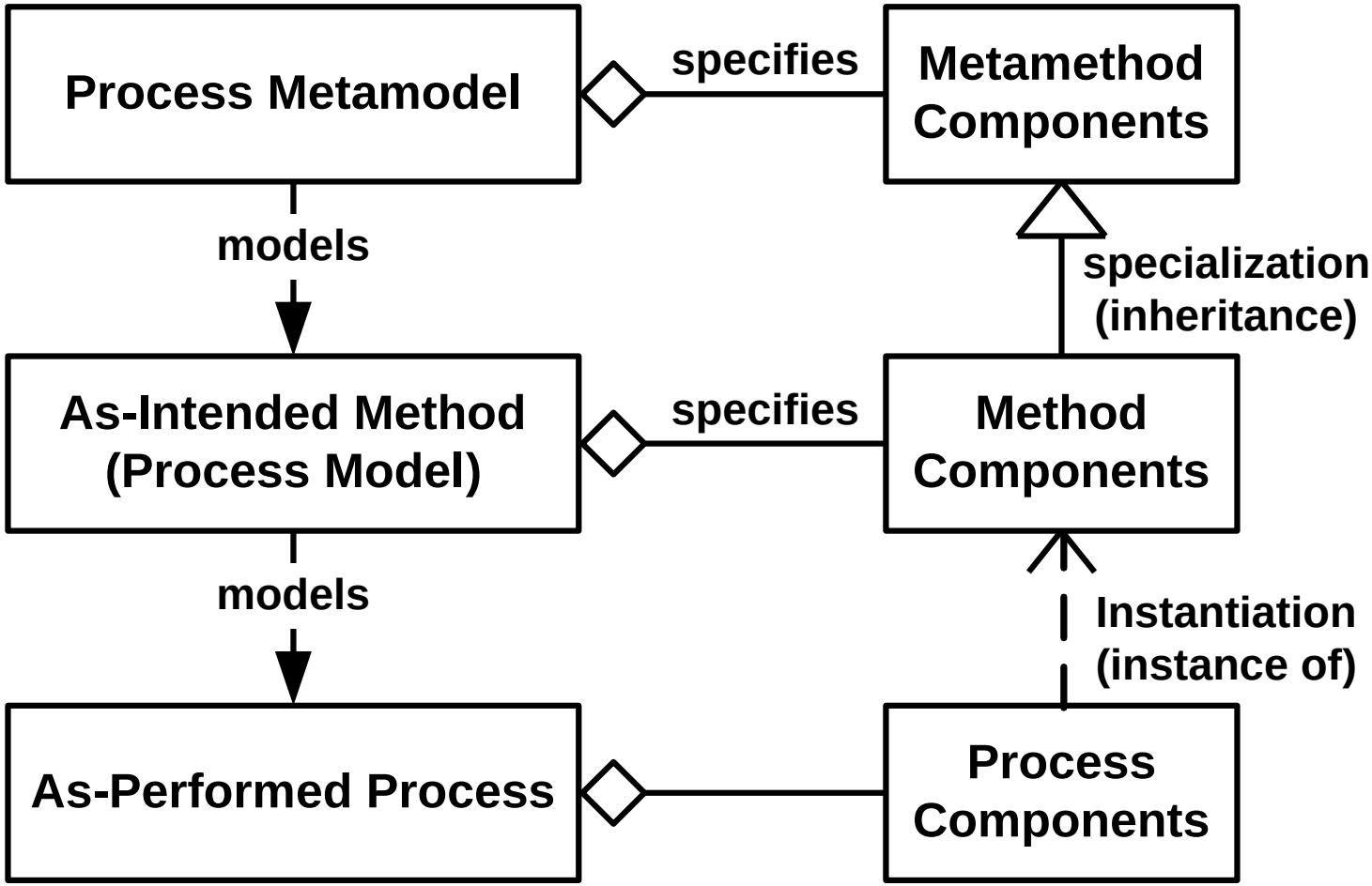
> an *actual* way that system architecture engineering is performed in practice on an endeavor

Methods are models of processes.

Methods are enacted as processes.

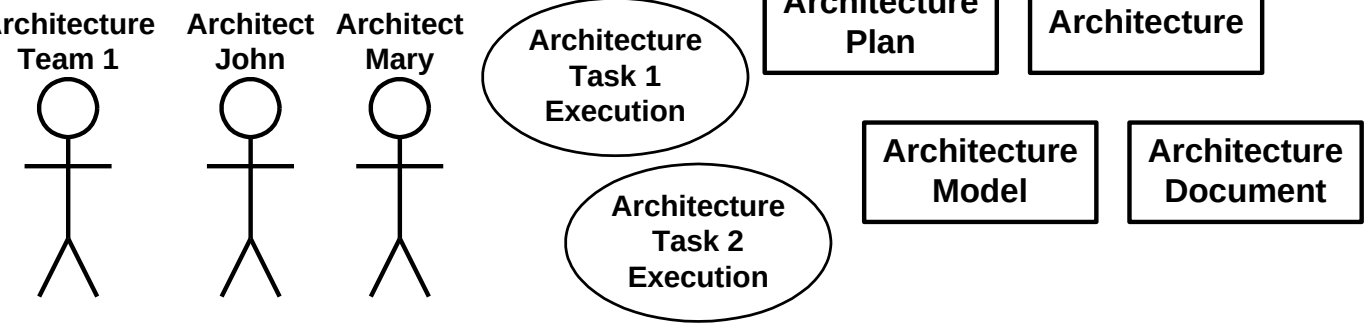
Method components are classes of process components.

Method Engineering Models

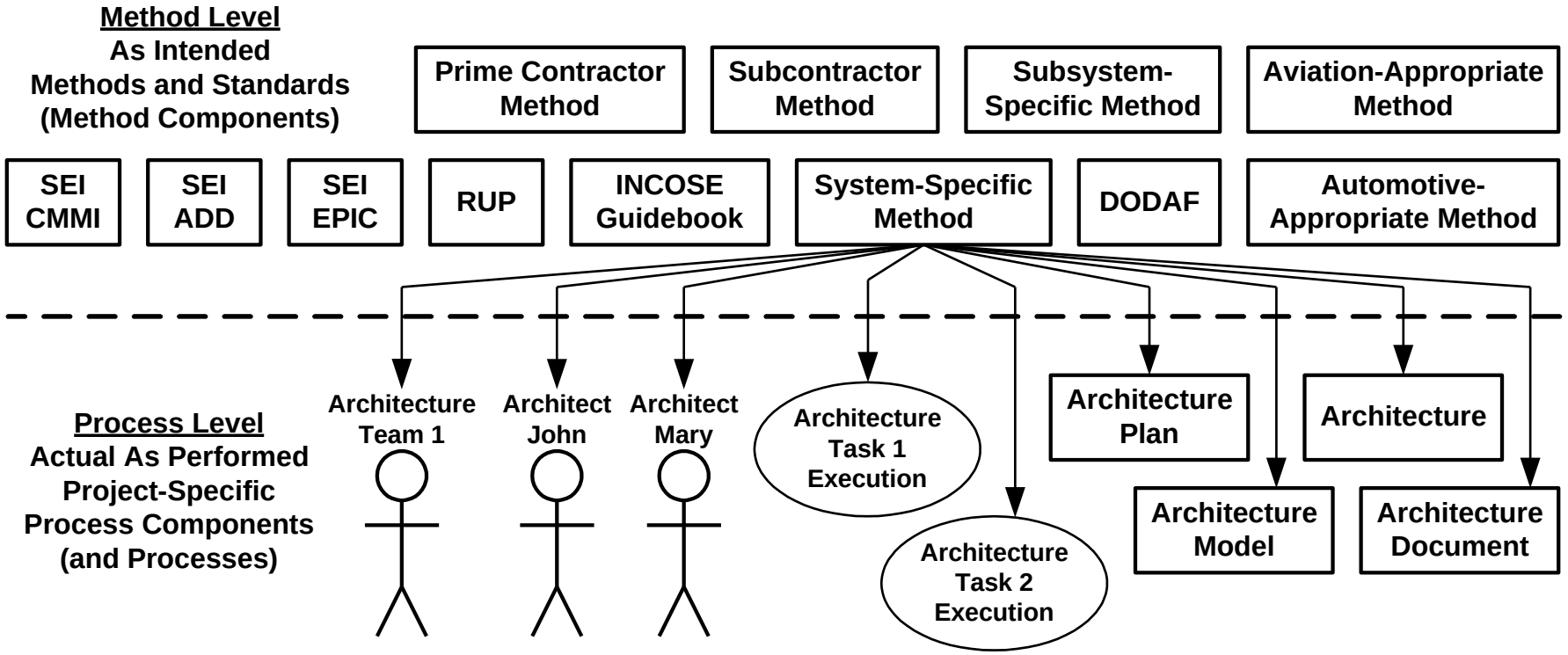


As-Performed Process Components

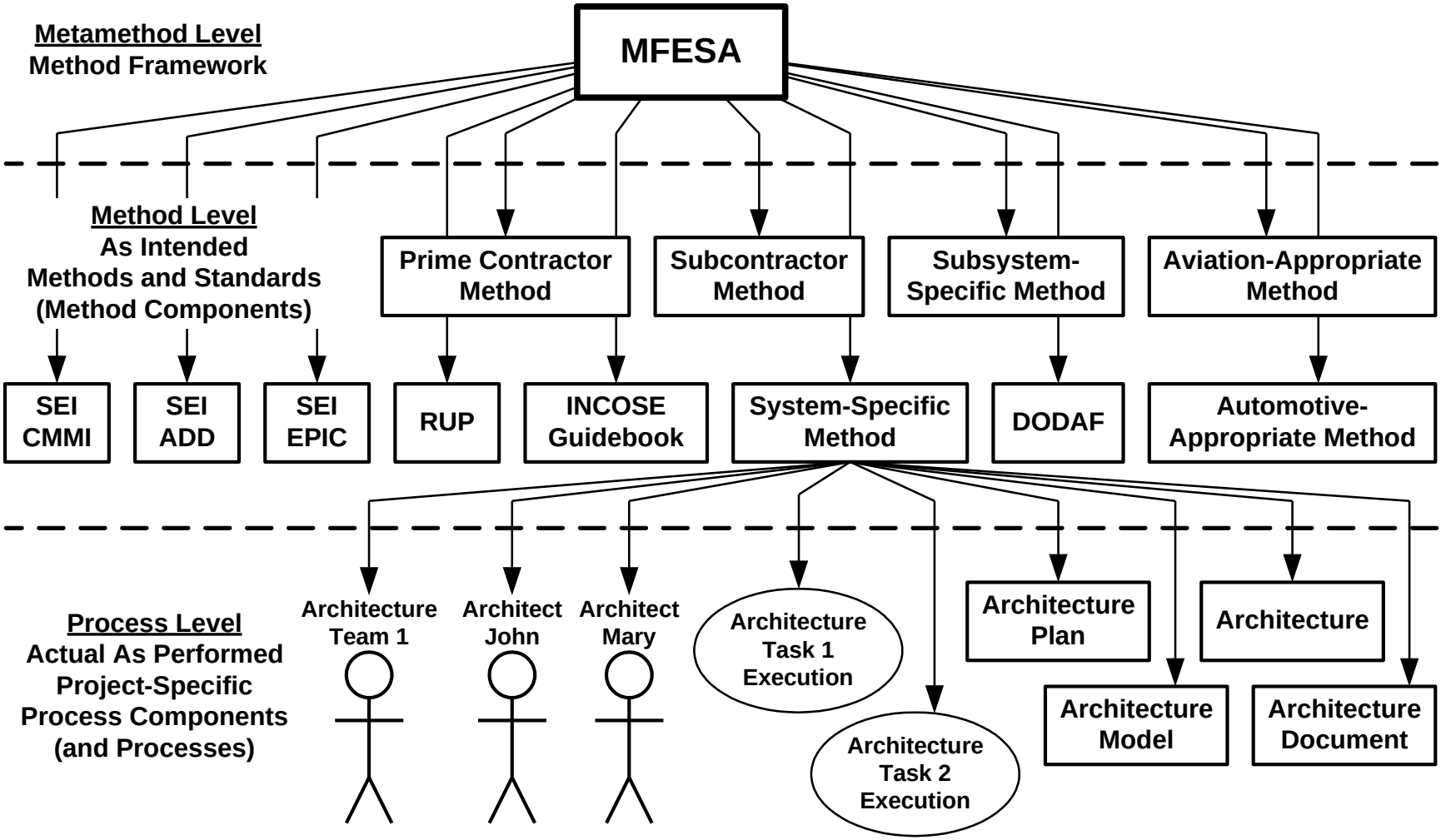
Process-Level
Actual As Performed Project-Specific Process Components (and Processes)



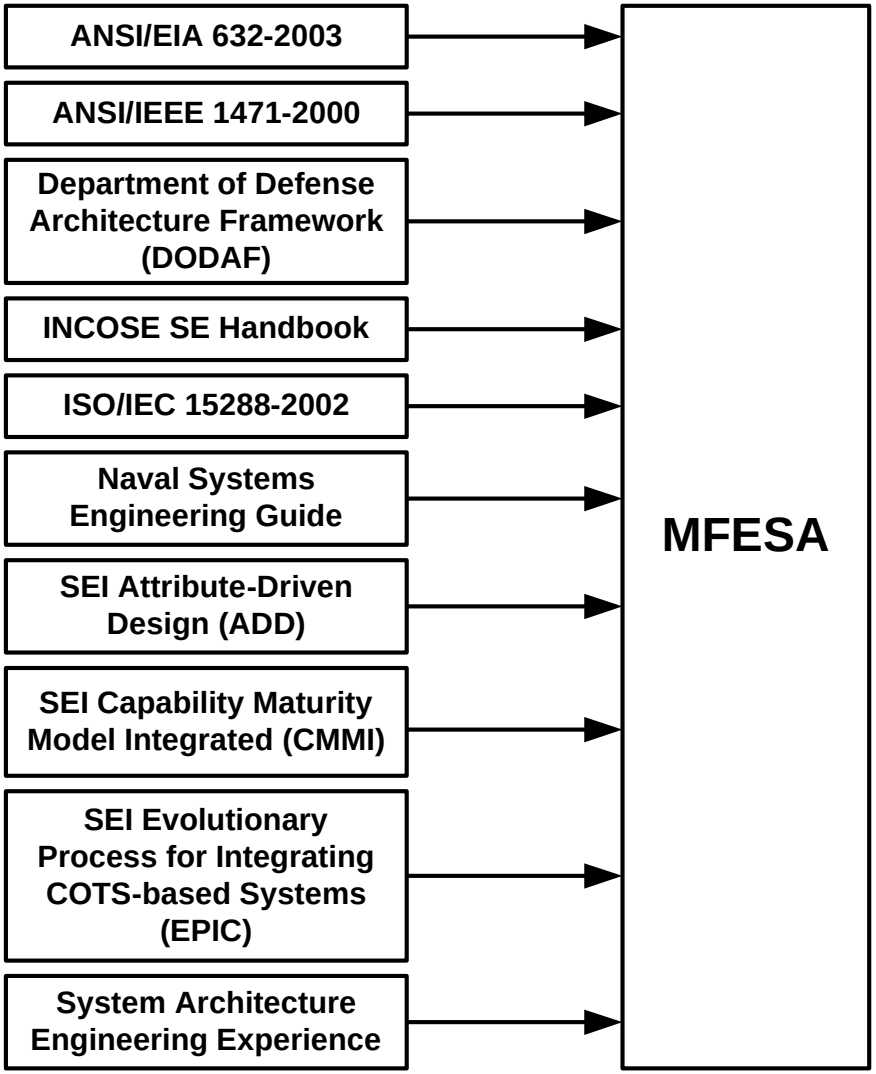
As-Intended Methods



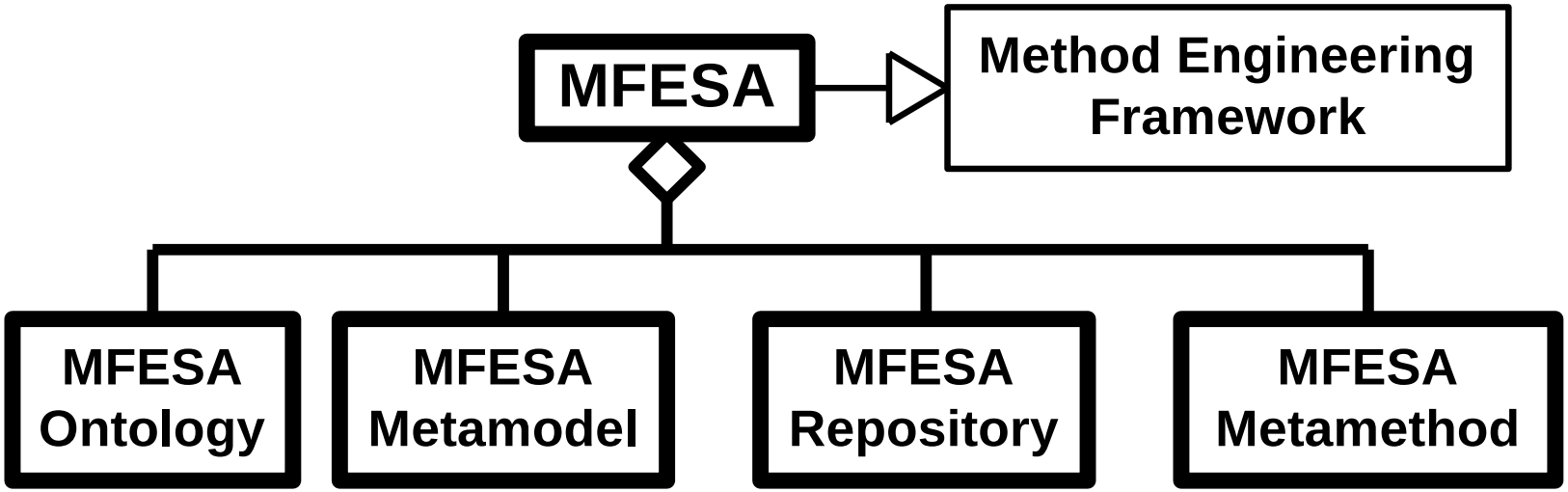
Method Frameworks



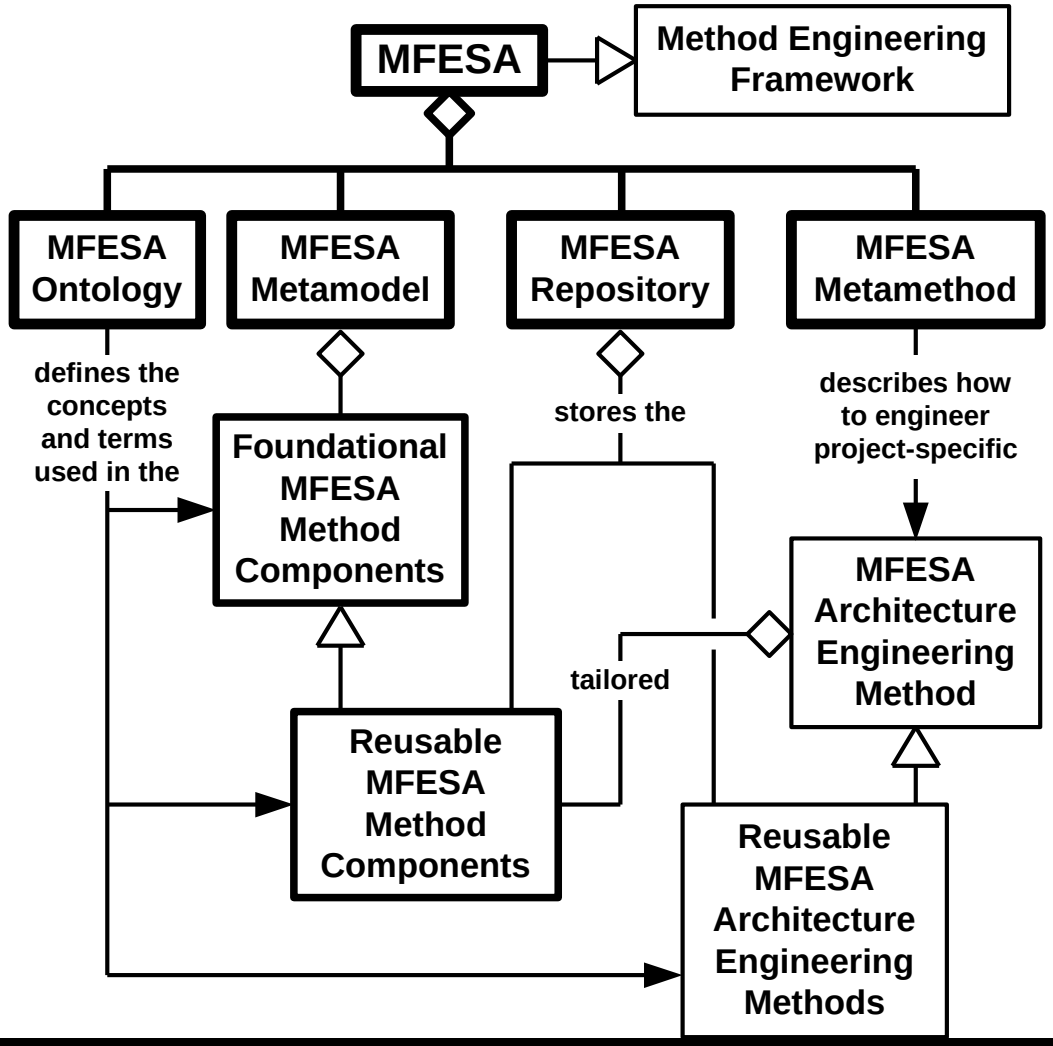
Primary Inputs to MFESA



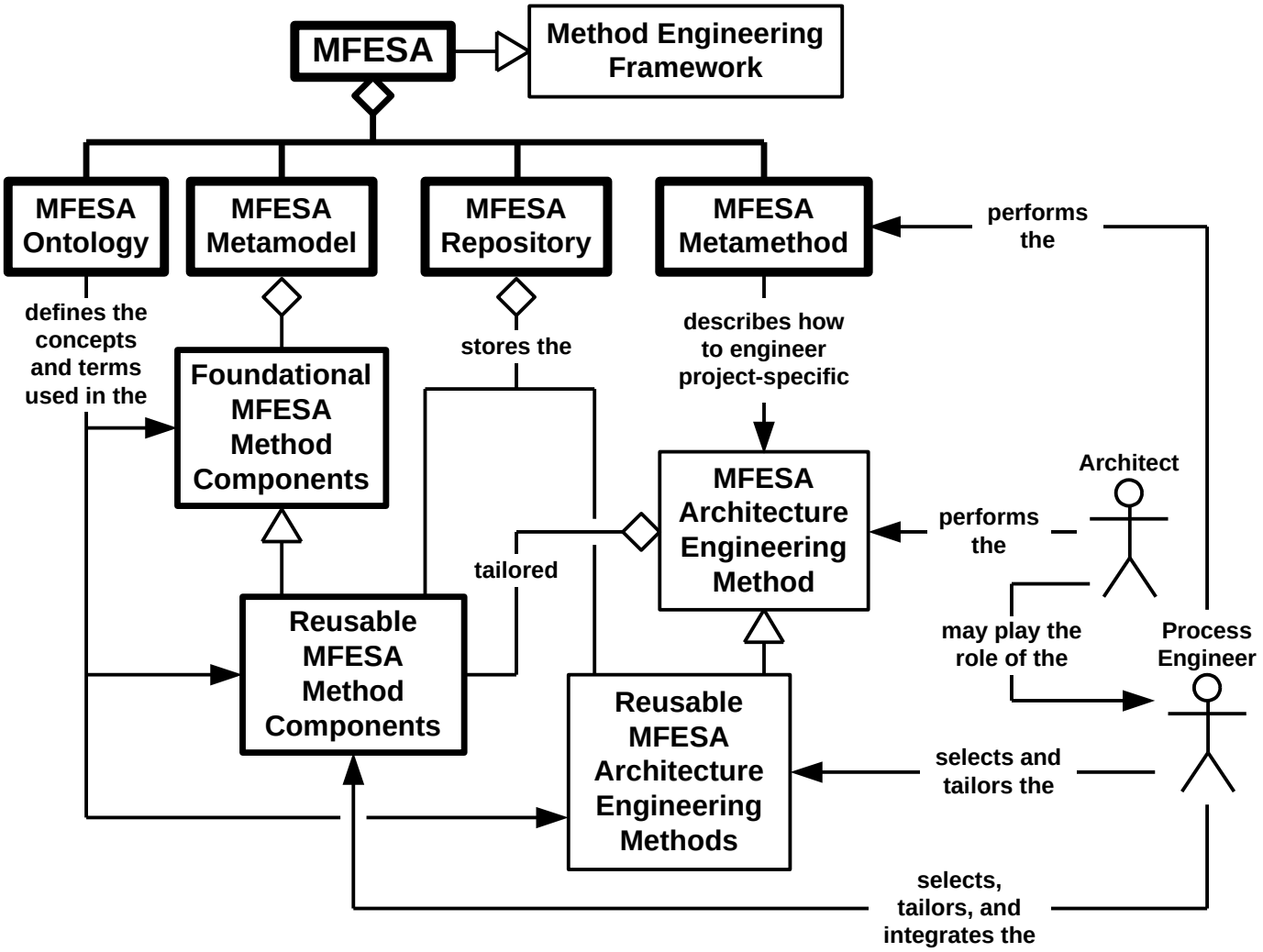
MFESA Components (Top View)



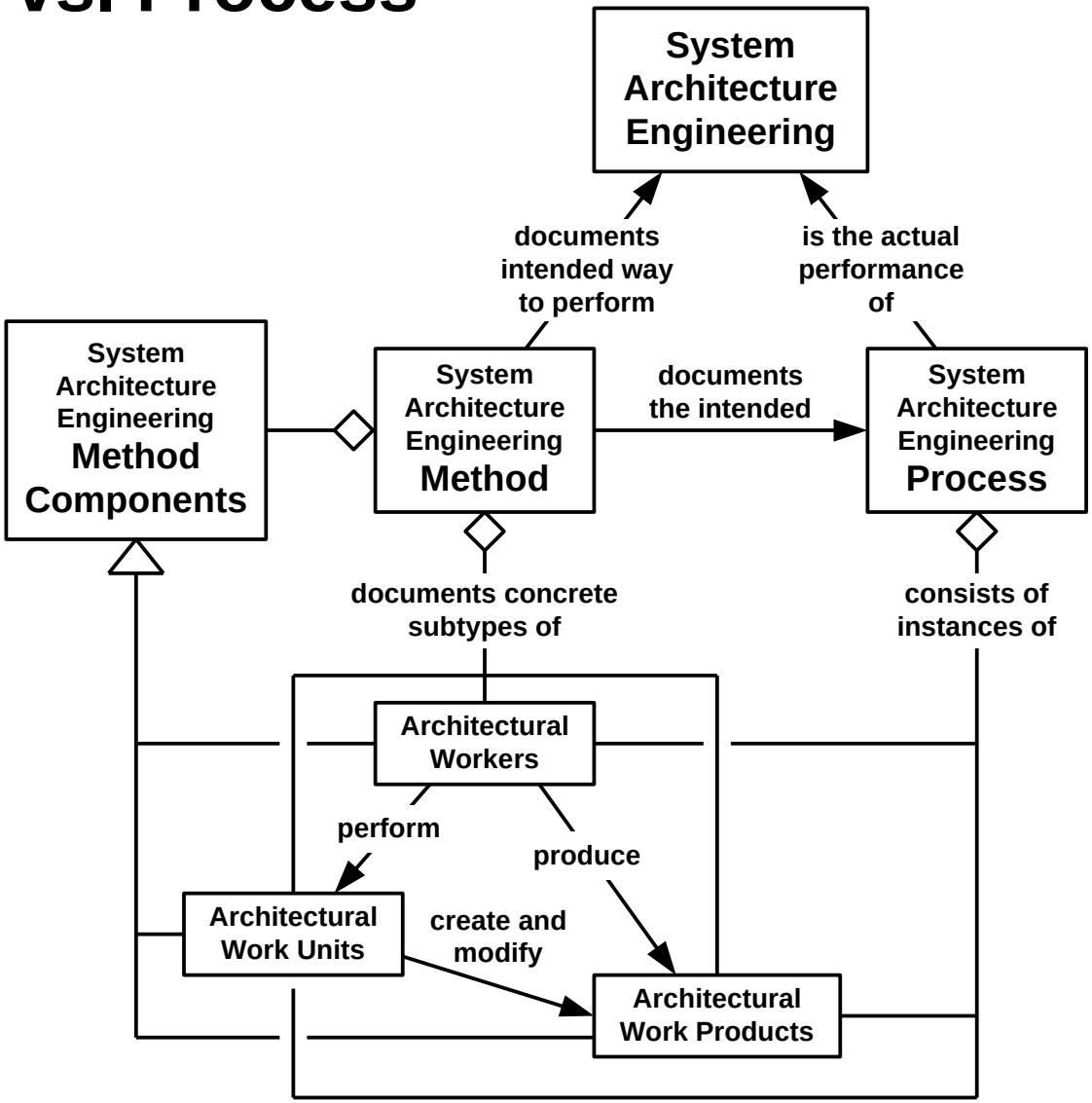
MFESA Components (Detailed View)



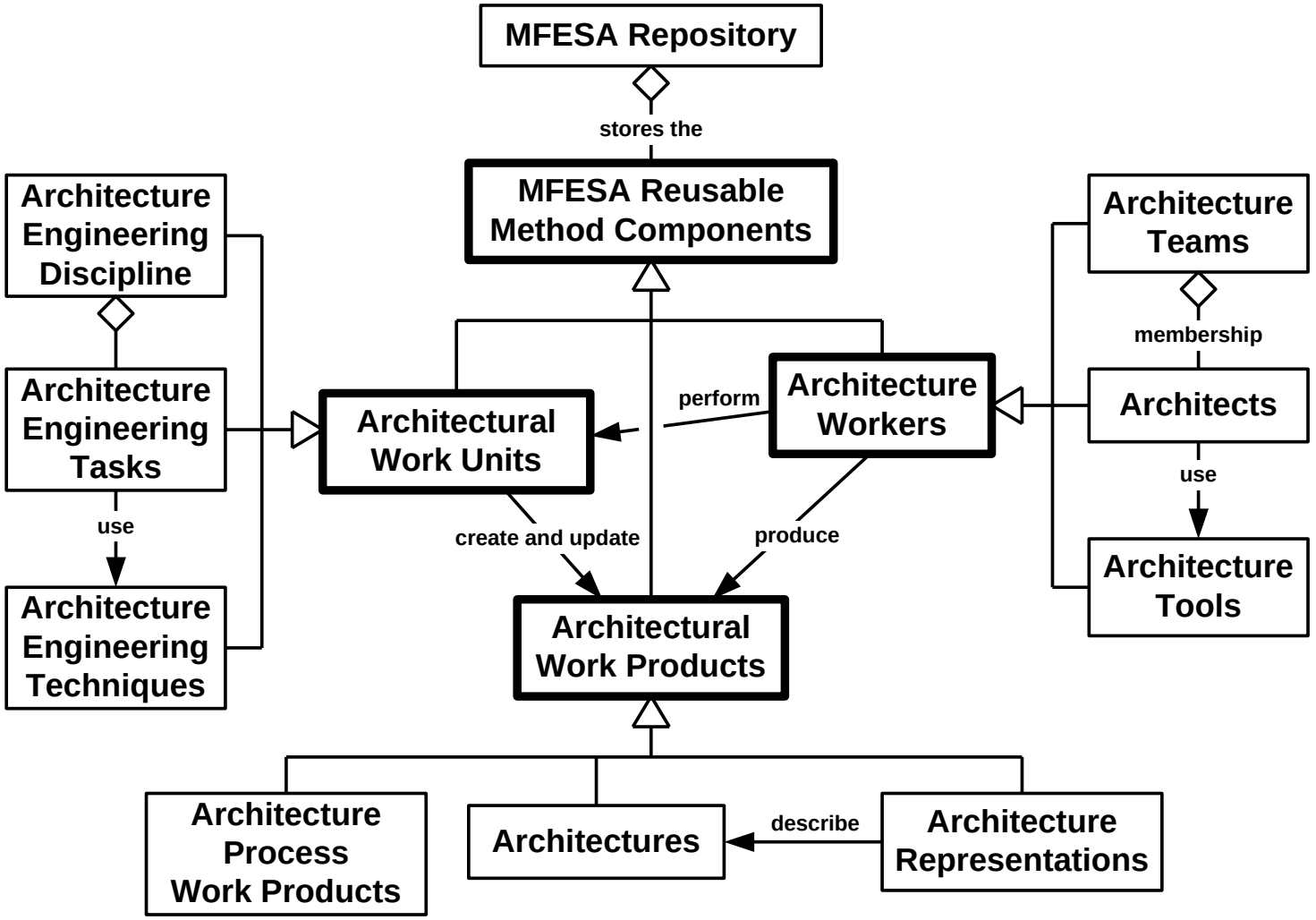
MFESA Components (Usage)



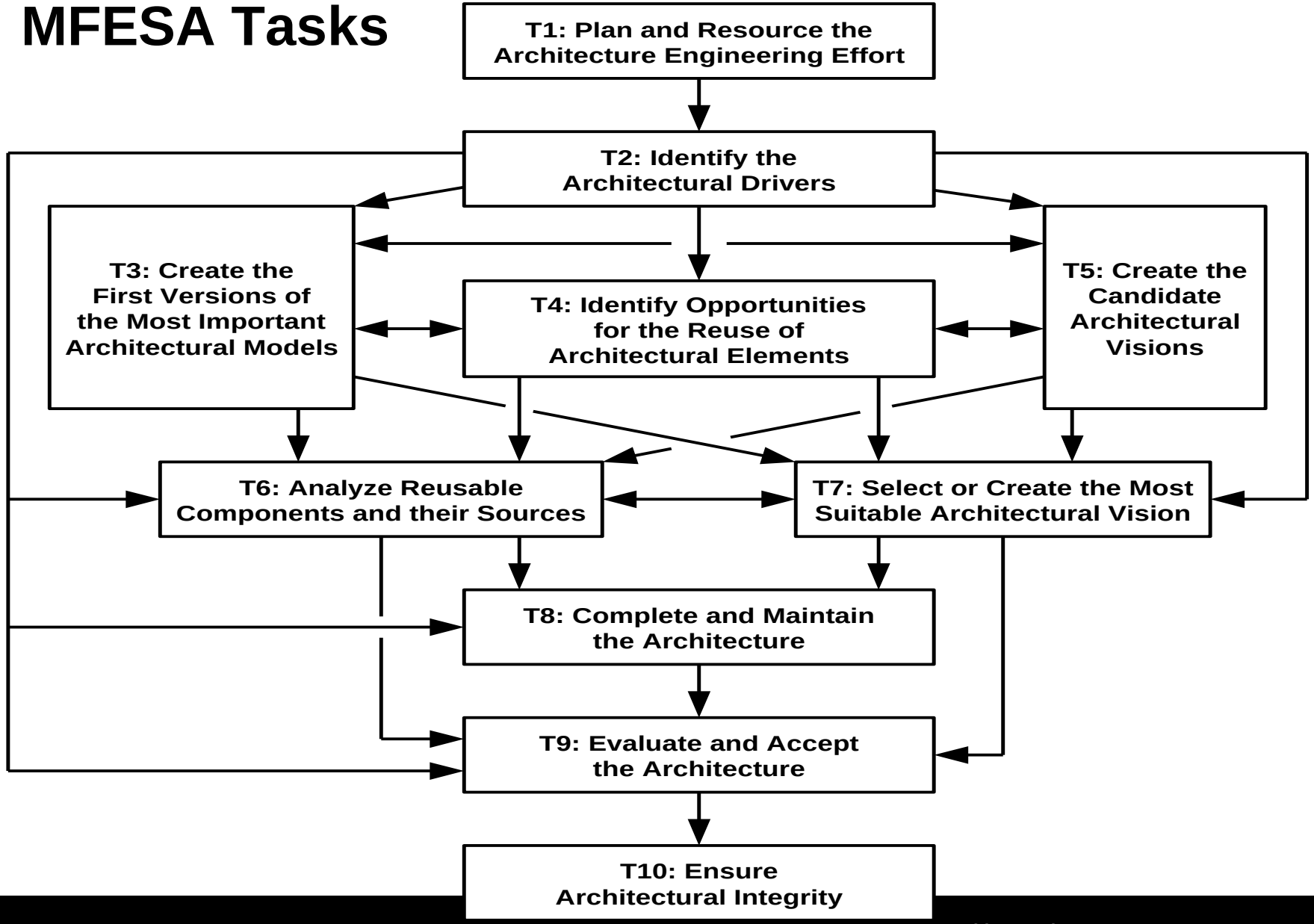
Method vs. Process



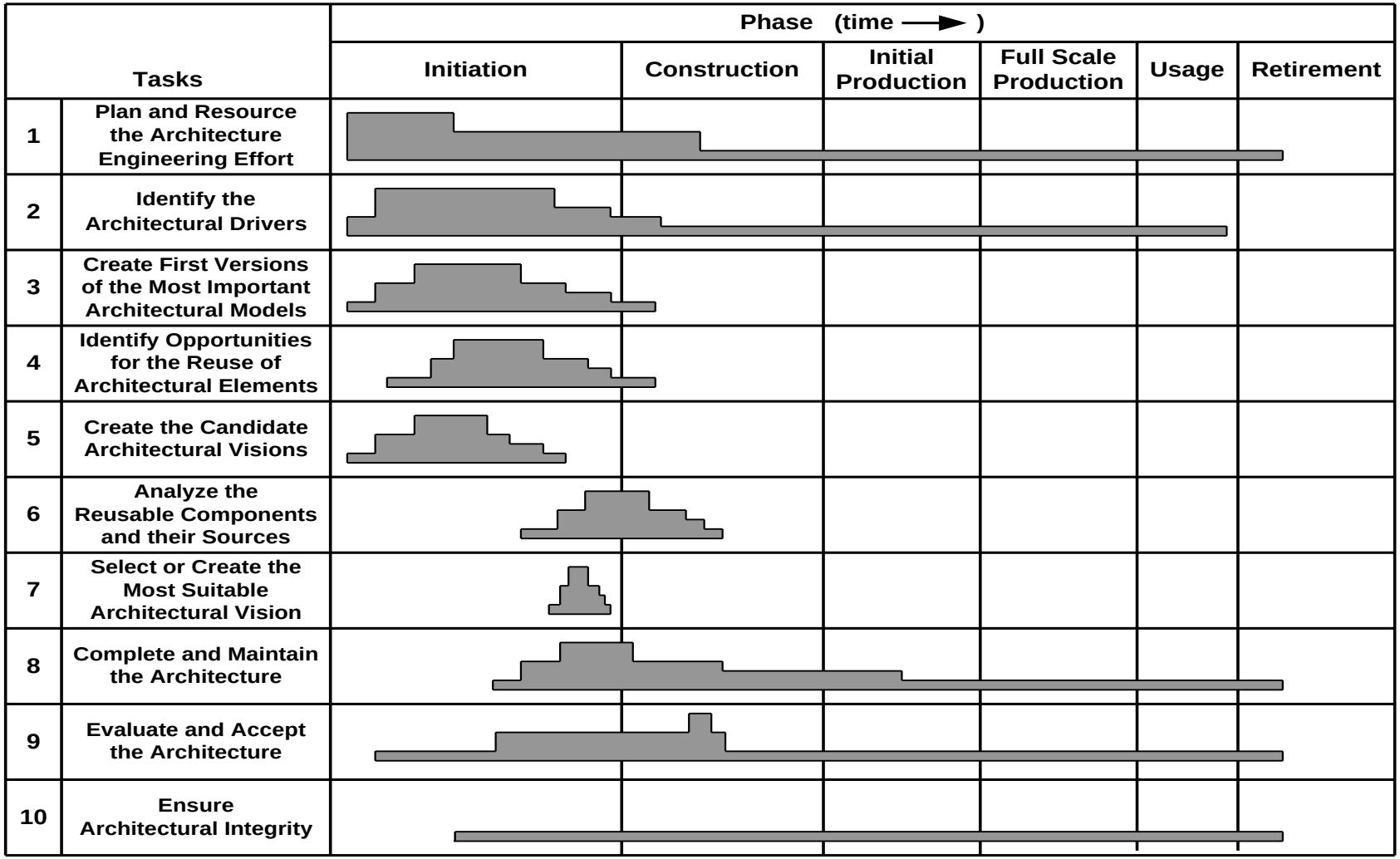
MFESA Metamodel of Reusable Method Components



MFESA Tasks



Effort by MFESA Task



MFESA Task 1) Plan and Resource the Architecture Engineering Effort

Goal:

- Prepare the system engineering team(s) 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(s).
- Develop plans, standards, and procedures for engineering the system architecture.
- Prioritize and schedule the system architecture engineering effort.

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 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 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 associated 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 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 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 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 6) Analyze Reusable Components and their Sources

Goal:

- Determine if any existing architectural 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 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 8)
Complete and Maintain the Architecture

Goals:

- Complete the 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 architecture.
- 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 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 if justified by 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 10)
Ensure Architectural Integrity

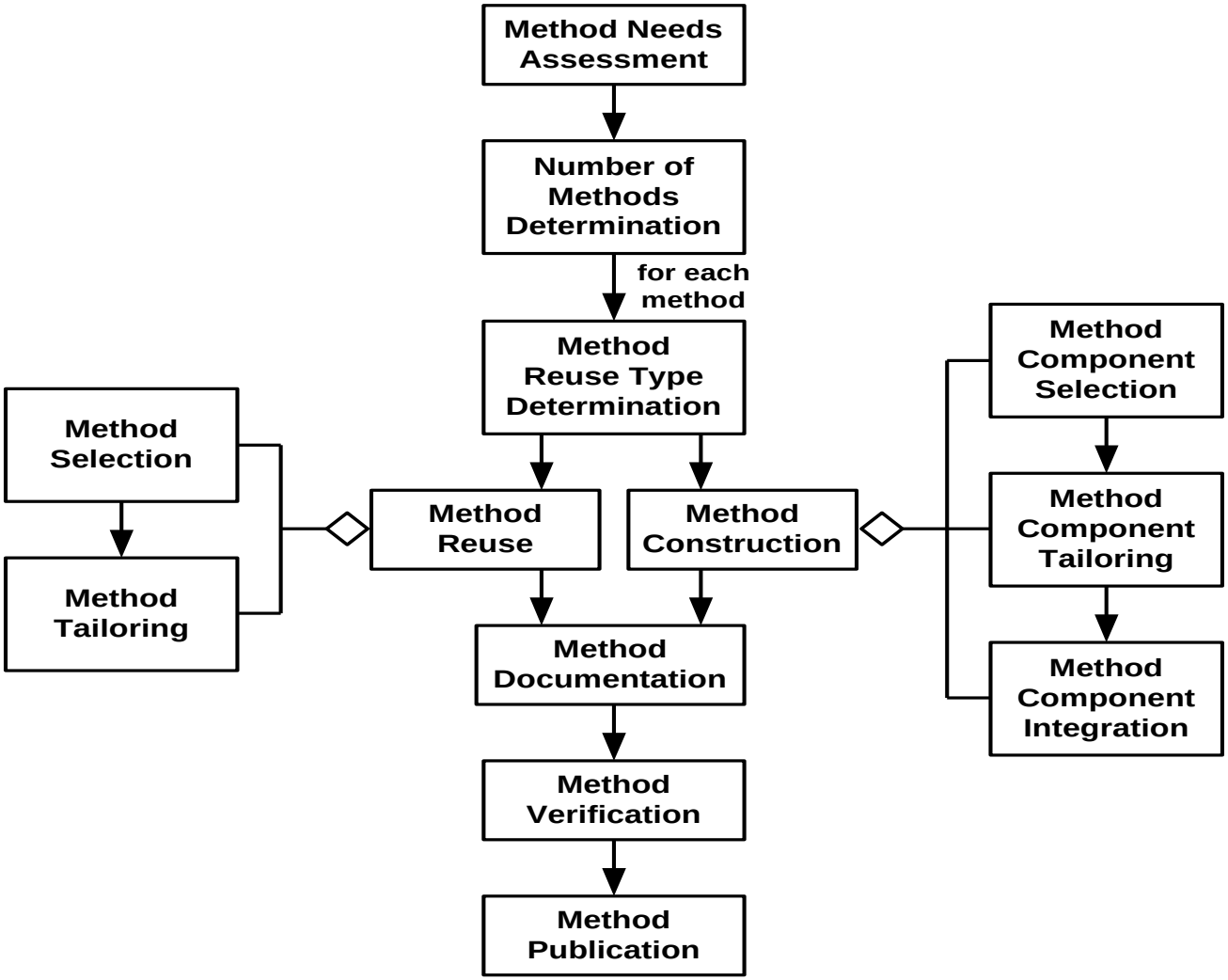
Goal:

- 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 the:
  - Architecturally Significant Requirements
  - Enterprise Architecture(s)
  - Reference Architecture(s)
  - Design of architectural components
  - Implementation of architectural components
- Ensure that the system architecture and its representations do not degrade over time.

MFESA Metamethod for Creating Appropriate Methods



Benefits of using MFESA

The benefits of:

- **Flexibility**: the resulting project/system-specific 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
- System architecture engineering(as-performed) processes.
- Architectures
- Architecture representations

To Obtain More Information

Book:

- http://www.amazon.com/Method-Framework-Engineering-System-Architectures/dp/1420085751/

Past Tutorial Slides:

- http://www.sei.cmu.edu/programs/acquisition-support/presentations/mfesa_tutorial_20080312.pdf

Conference Tutorials:

- IEEE Systems Conference 2009
  Vancouver, British Columbia, Canada
  23-24 March 2009 (All-Day Monday, 23 March)
- Systems and Software Technology Conference (SSTC)
  Salt Lake City, Utah, USA
  20-23 April 2009 (All-Day Monday, 20 April)

Contact Information Slide Format

Donald G. Firesmith
Senior Member Technical Staff
Acquisition Support Program
Telephone: +1 412-268-6874
Email: dgf@sei.cmu.edu

U.S. mail:
Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

World Wide Web:
www.sei.cmu.edu
www.sei.cmu.edu/contact.html

Customer Relations
Email: customer-relations@sei.cmu.edu
Telephone: +1 412-268-5800
SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257