



# ***Q*Uality Assessment of System Architectures and their *R*equirements (QUASAR)**

**DoD Systems Engineering  
Collaborator's Information  
Exchange (SECIE) Webinar**

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Donald Firesmith  
18 May 2010



# Topics

---

## History

Requirements and Architecture Challenges

Underlying Concepts

QUASAR Method

Key Lessons Learned



# History

---

F-35 Lightning II (Joint Strike Fighter) – 2003 to present

The JPO Air System Software Lead at the time (now an SEI VS) recognized the importance of system architecture.

Architecture assessments were not in contract, program plan, or schedule.

Concerned about cost and schedule, Lockheed Martin Aero (Prime Contractor) needed to be convinced to support assessments.

The JPO Air System Software Lead used compliance with architecturally-significant contract requirements as way to convince LM Aero to support assessments.

Prime contractor required agreement limiting scope of assessments (subsystems and quality characteristics).

Developed quality-case-based QUASAR method working jointly with JSF JPO and LM Aero Chief Architect.



# JSF JPO Recommendation

---

“I am writing to commend the CMU/SEI handbook called QUASAR [1], and its authors, principally Donald Firesmith. The F-35 Joint Strike Fighter (JSF) Program used it to assess the computer system architectures of our aircraft and ground systems. It helped us immensely in focusing attention on often-neglected quality attributes, rather than solely upon functional or component-based views of those systems. It guided us in both technical and managerial approaches to architecture assessment. QUASAR enabled the F-35 Program to verify fulfillment of its contractual architectural requirements, and in so doing, improve the quality of the product.

QUASAR's basis in CMU/SEI's real-world assessment experience, including on the F-35, undergirds its credibility and veracity. During the past four and one half years, F-35 used QUASAR to successfully assess major subsystems on nine occasions. I participated in the planning or execution of all these events, in my capacity as Mission Systems Architect, and later as Air System Architect. The handbook helped coordinate the efforts of the assessment teams (comprising the Program Office plus CMU/SEI and other subject matter experts) with system designers (comprising the air system contractor - Lockheed Martin, plus its suppliers).

I heartily recommend the continued use and development of this valuable tool.”

Mike Bossert, JSF JPO Mission Systems, 1 October 2009



# Topics

---

History

## Requirements and Architecture Challenges

Underlying Concepts

QUASAR Method

Key Lessons Learned



# Requirements and Architecture Challenges

---

Requirements and Architecture are the first two Opportunities to make Major Engineering Mistakes.

*Architecturally Significant* Requirements are typically poorly engineered.

Architecture and associated Architecturally Significant Requirements Affect:

- Project Organization and Staffing (Conway's Law)
- Downstream Design, Implementation, Integration, Testing, and Deployment Decisions

A common project-specific Quality Model is needed to drive the

- Quality Requirements, which drives the
- Quality of the System Architecture, which drives the
- Quality of the System



# Topics

---

History

Requirements and Architecture Challenges

## Underlying Concepts

QUASAR Method

Key Lessons Learned



# What is Quality?

---

## Quality

the Degree to which a Work Product (e.g., System, Subsystem, Requirements, Architecture) Exhibits a Desired or Required Amount of Useful or Needed Characteristics and Attributes

Not just lack of defects!

## Question:

What Types of Characteristics and Attributes are these?

## Answer:

They are the Characteristics defined by the Project Quality Model.



# Quality Model<sub>1</sub>

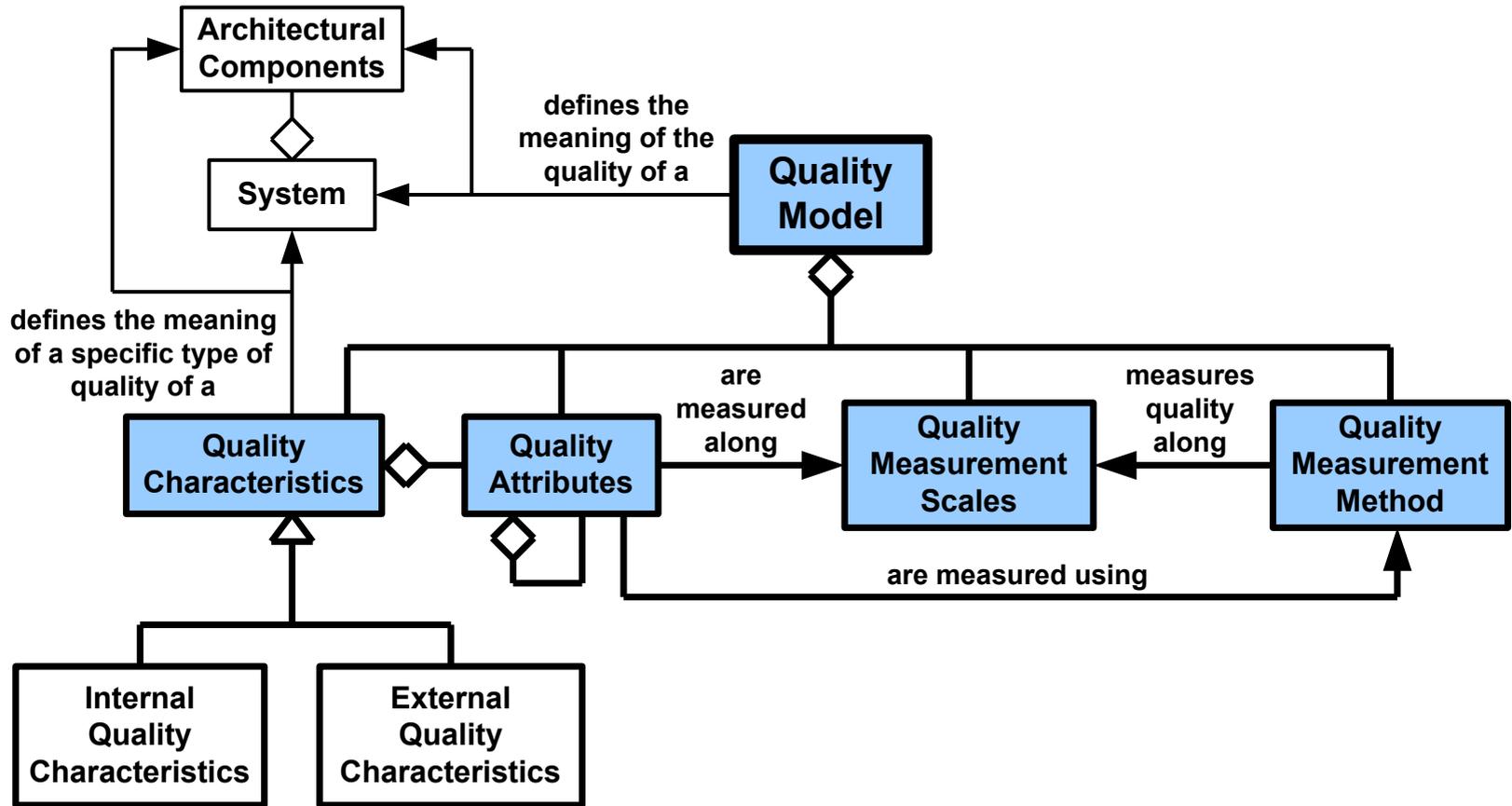
---

Quality of a System is defined in terms of a **Quality Model**:

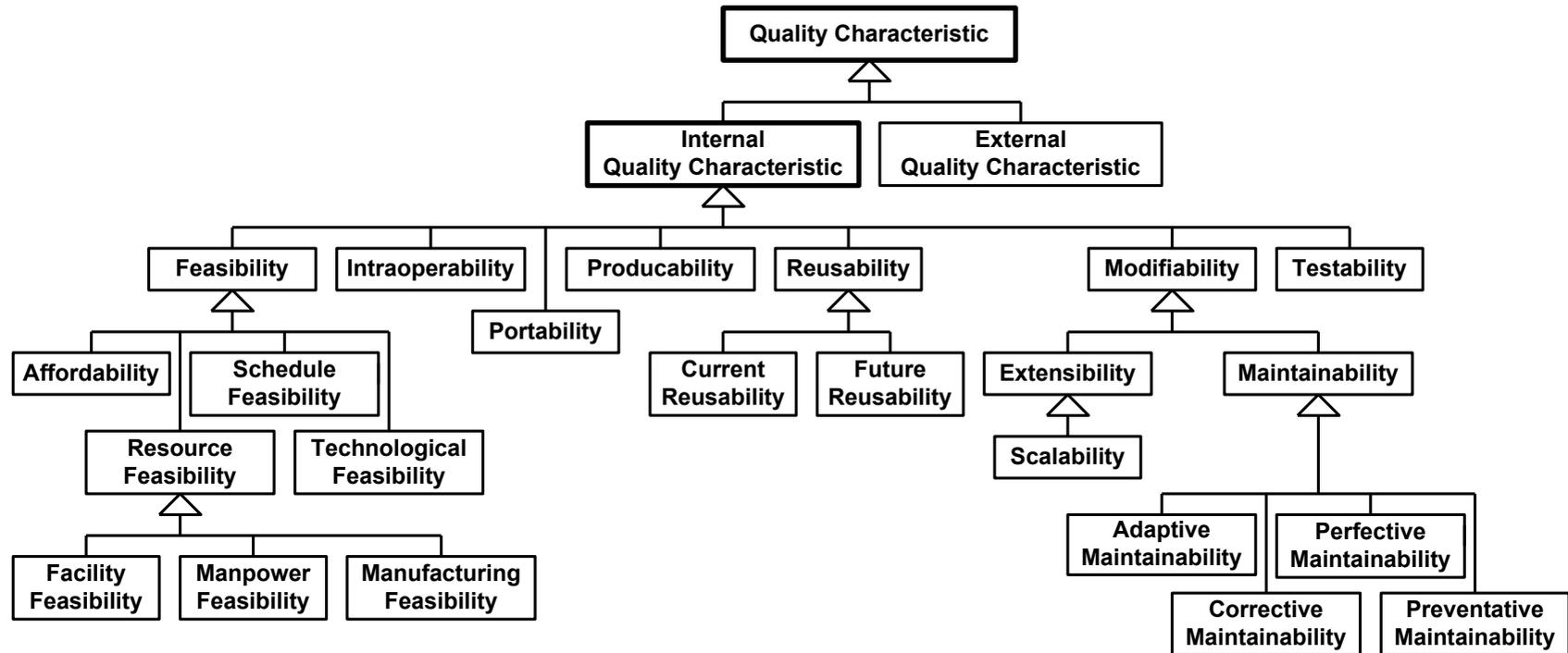
- **Quality Characteristics**  
(i.e., system-level characteristics also known as the ‘ilities’)  
(e.g., availability, extensibility, interoperability, maintainability, performance, portability, reliability, robustness, safety, security, survivability, and usability)
- **Quality Attributes**  
(e.g., the quality attributes of performance are jitter, latency, response time, schedulability, throughput)
- **Quality Measurement Scales**  
(e.g., milliseconds, transactions per second)
- **Quality Measurement Method**  
(e.g., operationally-defined test)



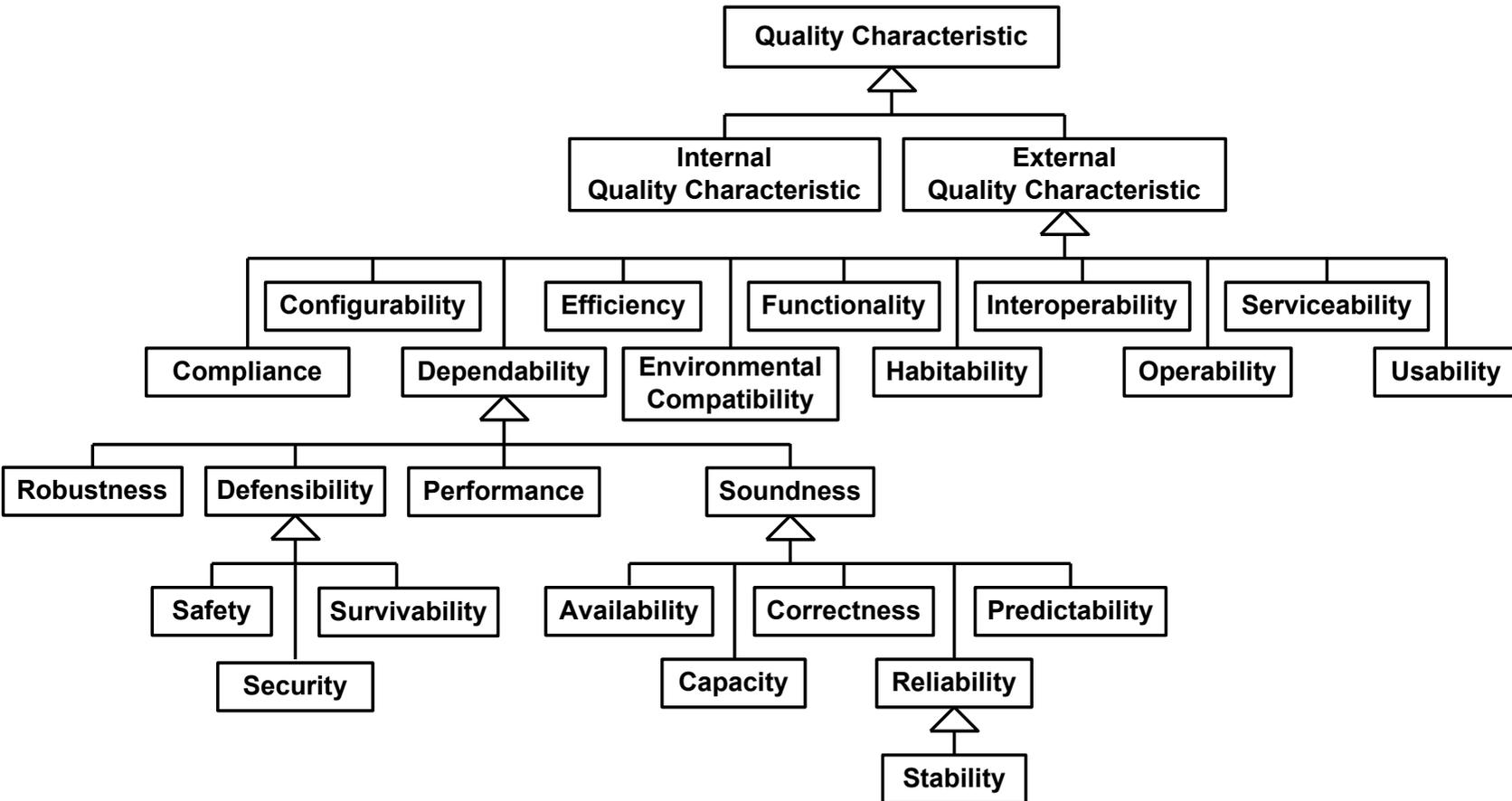
# Quality Model<sub>2</sub>



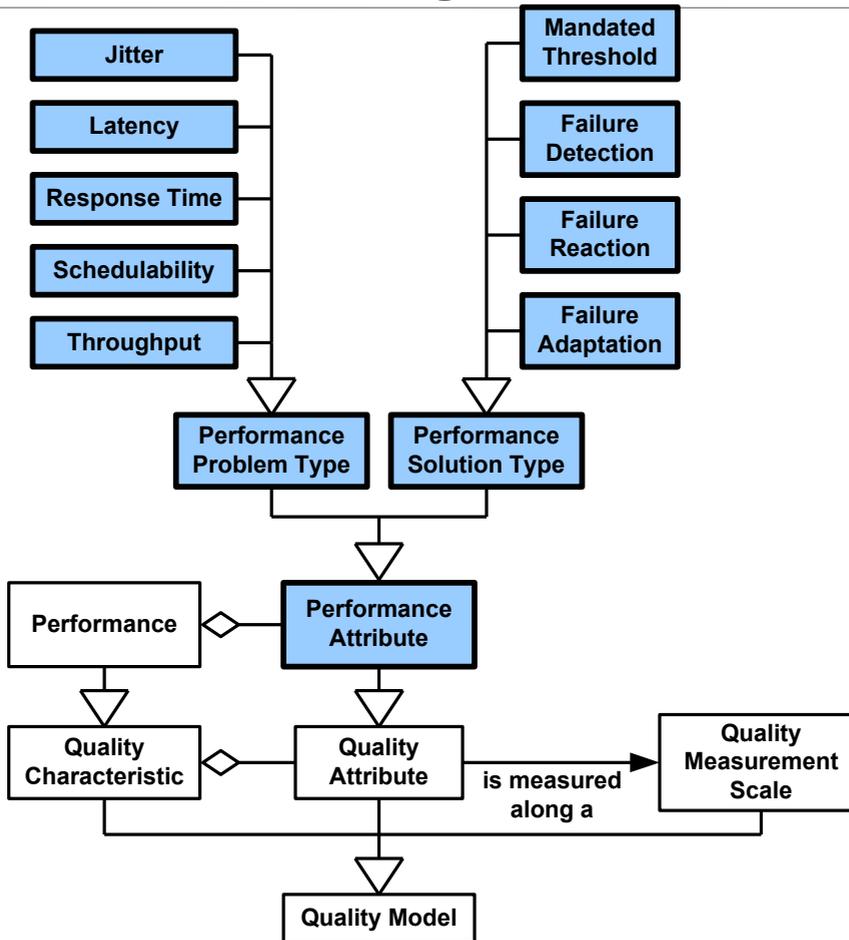
# Quality Model<sub>3</sub> – Internal Quality Characteristics



# Quality Model<sub>4</sub> – External Quality Characteristics



# Quality Model<sub>5</sub> – Performance Quality Attributes



# Quality Case - Definition

---

## Quality Case

a Cohesive Collection of *Claims, Arguments, and Evidence* that Makes the Developers' Case that their Work Product(s) have *Sufficient Quality*

## Foundational Concept underlying QUASAR

A Generalization and Specialization of Safety Cases from the Safety Community:

More) Can Address any Quality Characteristic and/or Quality Attribute  
Less) Often Restricted to only Requirements or Architecture

## Very similar to Assurance Cases:

Restricted to early in the lifecycle (requirements and architecture)  
Simplified arguments  
Quality cases can grow into full blown assurance cases.



# Quality Cases – Components<sub>1</sub>

---

A Quality Case consists of the following types of Components:

## 1. Claims

Developers' Claims that their Work Products have *Sufficient* Quality, whereby quality is defined in terms of the quality characteristics and quality attributes defined in the official project quality model

## 2. Arguments

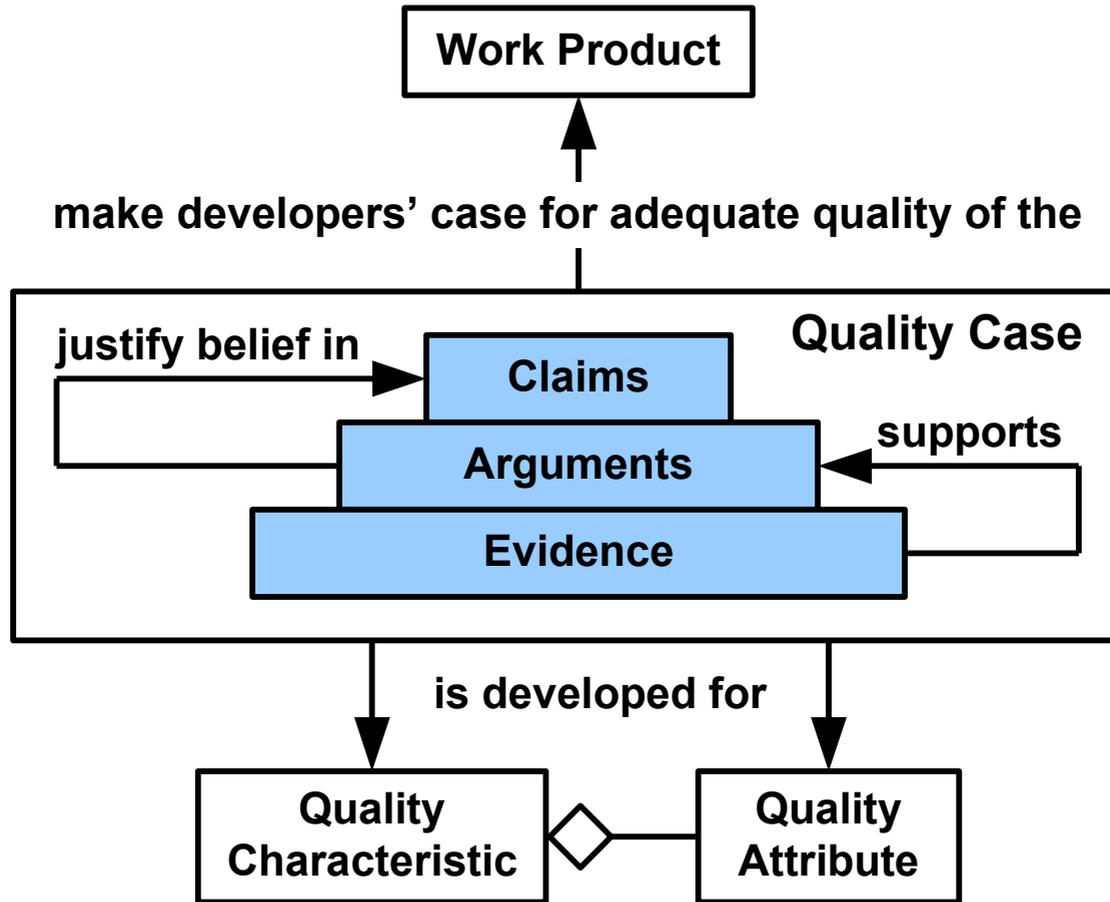
Clear, Compelling, and Relevant Developer Arguments Justifying the Assessors' Belief in the Developers' Claims  
(e.g., decisions, inventions, trade-offs, analysis and simulation results, assumptions, and associated rationales)

## 3. Evidence

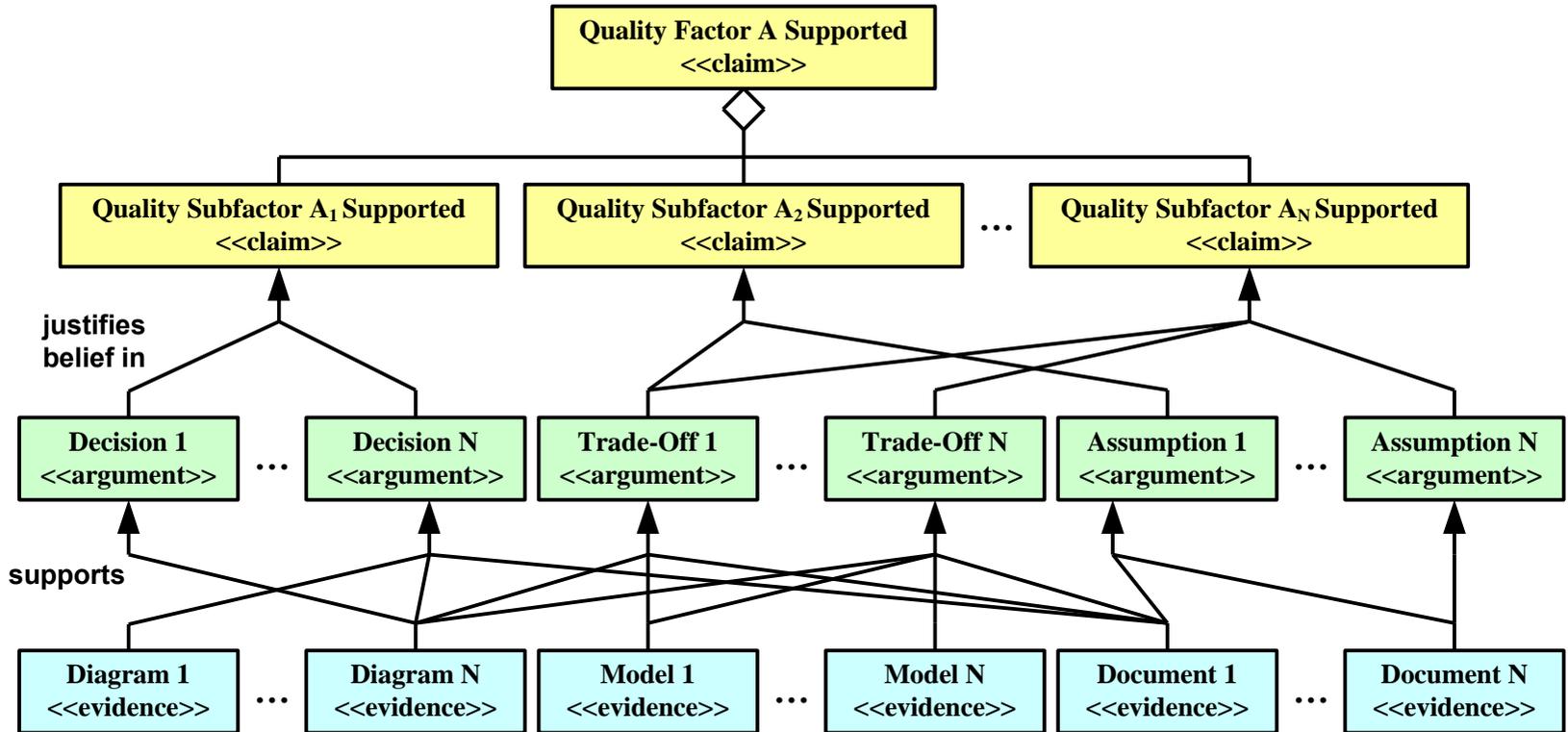
Adequate Credible Evidence Supporting the Developers' Arguments  
(e.g., official project diagrams, models, requirements specifications and architecture documents; requirements repositories; analysis and simulation reports; test results; and demonstrations witnessed by the assessors)



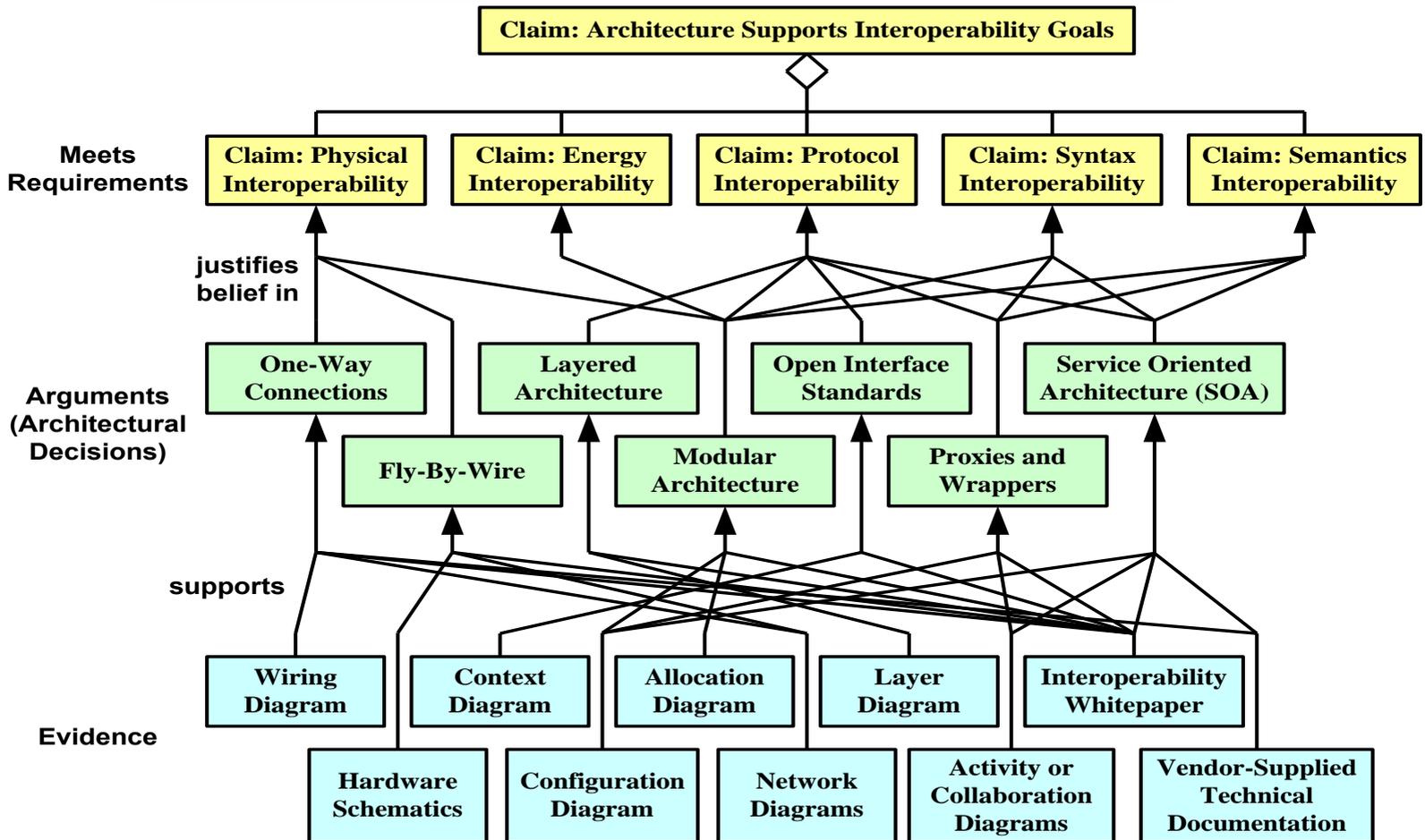
# Quality Cases – Components<sub>2</sub>



# Quality Case Diagram Notation



# Architectural Interoperability Case Diagram



# Specialized QUASAR Quality Cases

---

QUASAR utilizes the following specialized types of Quality Cases:

1. Requirements Quality Cases
2. Architectural Quality Cases

QUASAR Version 1 only had Architectural Quality Cases.

QUASAR Versions 2 and 3 have Both Types of Quality Cases.



# QUASAR Quality Case Responsibilities

---

## Requirements Engineers and Architects' Responsibilities:

- Prepare Quality Cases
- Provide Preparation Materials (including Presentation Materials and Quality Cases) to Assessors Prior to Assessment Meetings
- Present Quality Cases (Make their Case to the Assessors)
- Answer Assessors' Questions

## Assessor Responsibilities:

- Prepare for Assessments
- *Actively* Probe Quality Cases
- Develop Consensus regarding Assessment Results
- Determine and Report Assessment Results:
  - Present Outbriefs
  - Publish Reports



# What is a System?

---

## System

a Major, Cohesive, Executable, and Integrated Set of *Architectural Elements* that Collaborate to Provide the Capability to Perform one or more related *Missions*

## Systems are Decomposed into Architectural Components:

- Subsystems
- Data
- Documentation
- Hardware
- Software
- Manual Procedures
- Personnel (e.g., Roles such as Operators and Administrators)
- Equipment, Facilities, Materials, and Tools



# Systems Imply

---

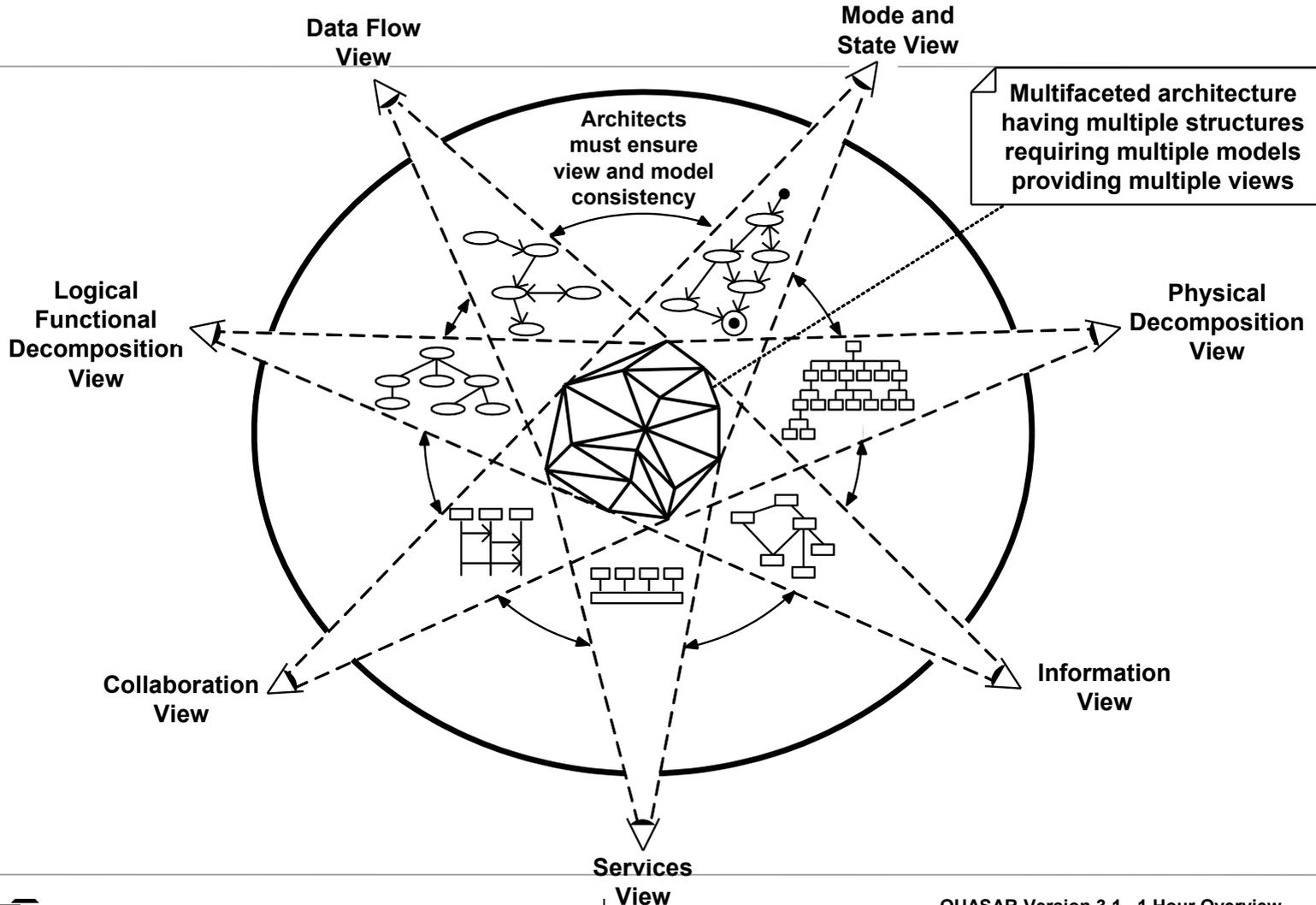
Multiple Static and Dynamic Logical and Physical “Structures” that exist at Multiple ‘Tiers’ in the System:

- Static Functional Decomposition Logical Structure
- Static Subsystem Decomposition Physical Structure
- Hardware, Software, and Data Structures
- Allocation Structure (Software and Data to Hardware)
- Network Structure
- Concurrency (Process) Structure

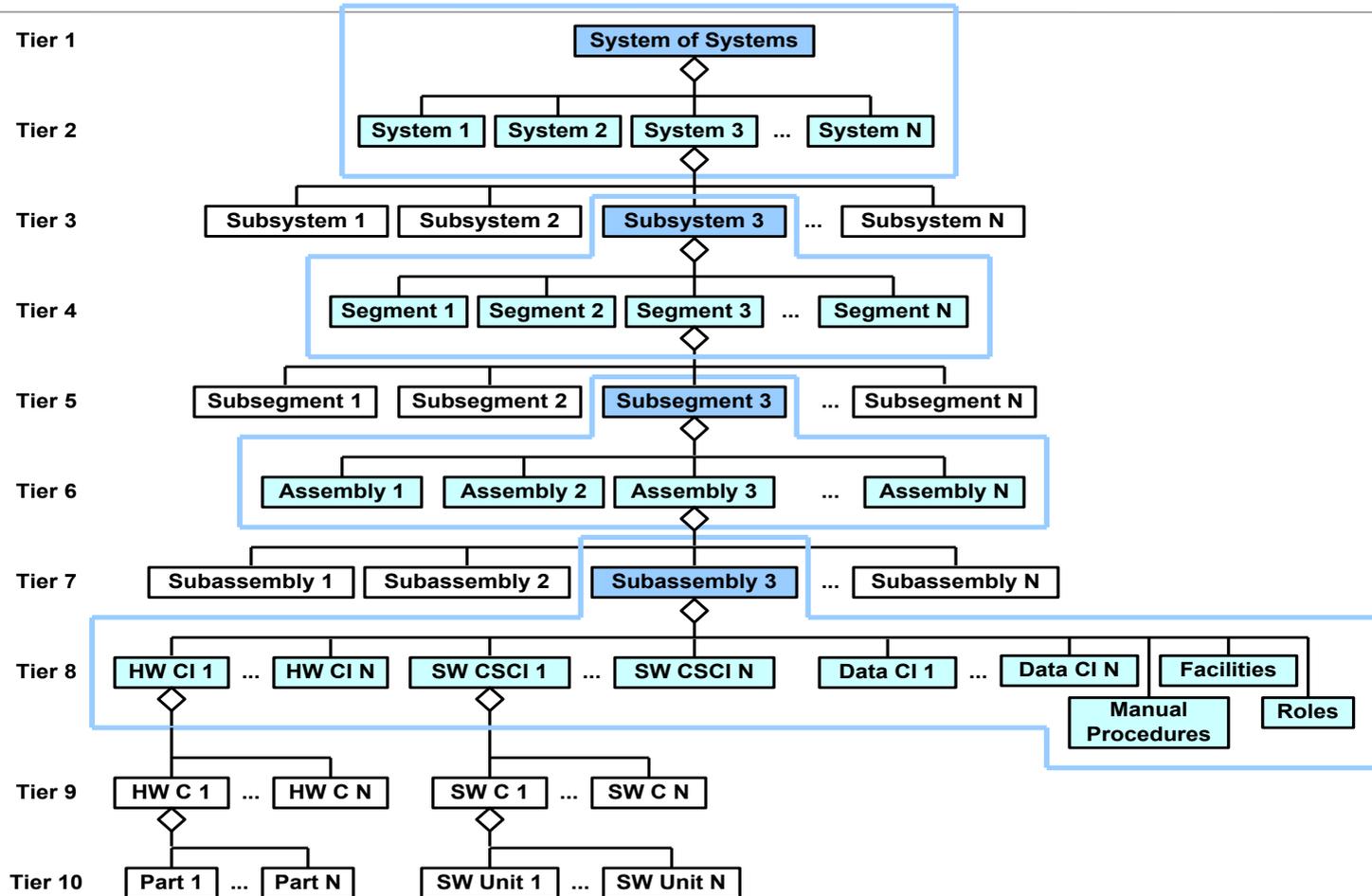
Multiple Specialty Engineering Focus Areas  
(e.g., Performance, Reliability, Safety, and Security)



# Some Example Views of Models of Structures



# Example QUASAR Scope – Four Assessments



# What is a System Architecture?<sub>1</sub>

---

## System Architecture

*the Most Important, Pervasive, Top-Level, Strategic Decisions, Inventions, Engineering Trade-Offs, Assumptions, and associated Rationales about How a System's Architectural Elements will collaborate to meet the System's Derived and Allocated Requirements*

More than just structure!



# What is a System Architecture?₂

---

System Architecture Includes:

- **The System's Numerous Static and Dynamic, Logical and Physical Structures**  
(i.e., Essential Architectural Elements, their Relationships, their Associated Blackbox Characteristics and Behavior, and how they Collaborate to Support the System's Mission and Requirements)
- **Architectural Decisions, Inventions, and Tradeoffs**  
(e.g., Styles, Patterns, and Mechanisms used to ensure that the System Achieves its Architecturally-Significant Product and Process Requirements, especially the Quality Requirements or 'ilities')
- **Strategic and Pervasive Design-Level Decisions**  
(e.g., using a *Design* Paradigm such as Object-Orientation or Mandated Widespread use of common Design Patterns)
- **Strategic and Pervasive Implementation-Level Decisions**  
(e.g., using a Safe Subset of C++)



# Architecturally Significant Requirements

---

## Architecturally Significant Requirements

any Requirement that has a Significant Impact on a System / Subsystem Architecture

Architecturally Significant Requirements typically include:

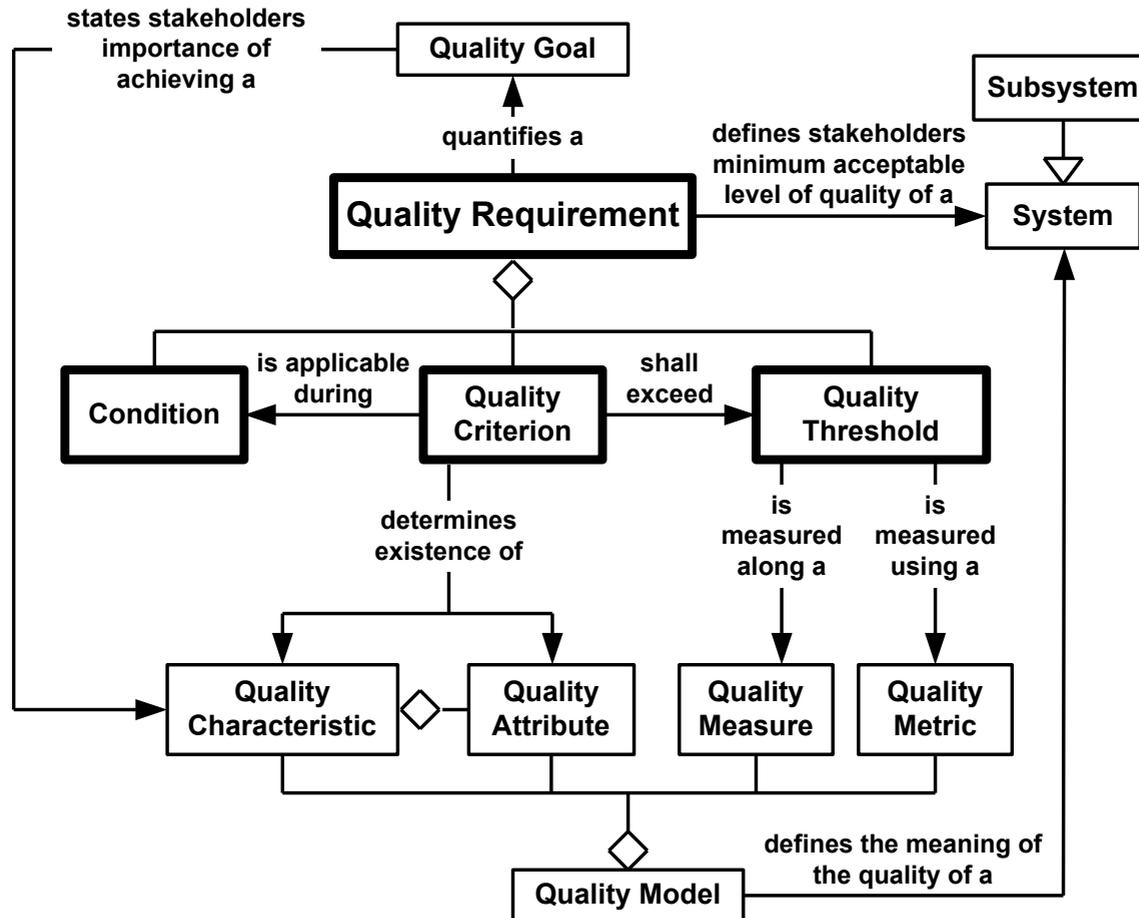
- Quality Requirements, which specify Minimum Amounts of some Quality Attribute or Characteristic
- Architectural Constraints
- Primary Mission Functional Requirements (Feature Sets)

Quality Requirements are often the:

- Most Important
- Least Well Engineered



# Quality Requirements – Components



# Topics

---

History

Requirements and Architecture Challenges

Underlying Concepts

**QUASAR Method**

Key Lessons Learned



# Definition

---

## Quality Assessment of System Architectures and their Requirements

a Well-Documented and Proven Method based on the use of *Quality Cases* for *Independently* Assessing the *Quality* of:

- Software-intensive *System / Subsystem Architectures* and the
- *Architecturally Significant Requirements* that Drive Them



# QUASAR Philosophy<sub>1</sub>

---

## Requirements Engineers (REs) should *Make Case* to Assessors:

- REs *should* know Stakeholder Needs and Goals
- REs *should* know What they Did and Why  
(Architecturally-Significant Requirements, Rationales, & Assumptions)
- REs *should* Know Where they Documented the Architecturally-Significant Requirements in their Work Products

## Architects should *Make Case* to Assessors:

- Architects *should* know the Architecturally-Significant Requirements
- Architects *should* know What they Did and Why  
(Decisions, Inventions, Trade-Offs, Assumptions, and Rationales)
- Architects *should* know Where they Documented their Architectural Work Products



# QUASAR Philosophy<sub>2</sub>

---

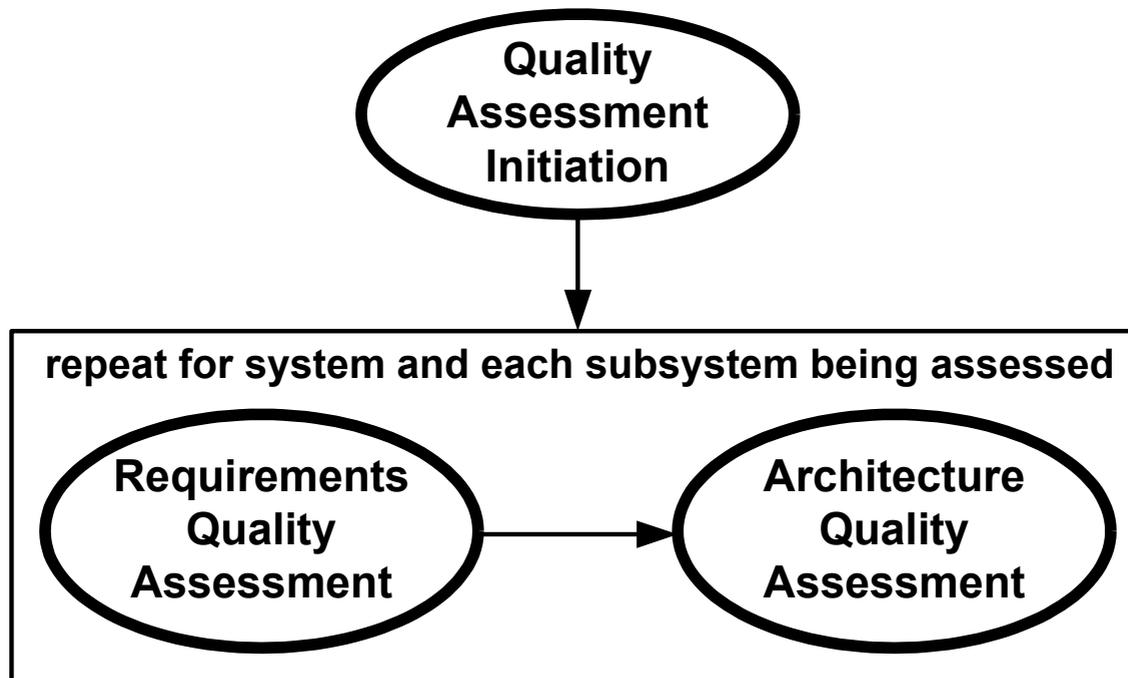
**Assessors** should *Actively* Probe Quality Cases:

- **Claims Correct and Complete?**  
Do the Claims include *all* relevant Quality Characteristics, Quality Attributes, Quality Goals, and Quality Requirements?
- **Arguments Correct, Complete, Clear, and Compelling?**  
Do the Arguments include *all* relevant Quality Characteristics, Quality Attributes, Quality Goals, Quality Requirements, Decisions, Inventions, Trade-offs, Assumptions, and Rationales?
- **Arguments Sufficient?**  
Are the Arguments Sufficient to Justify the Claims?
- **Evidence Sufficient?**  
Is the Evidence Sufficient to Support the Arguments?
- **Current Point in the Schedule?**  
Are the Claims, Arguments, and Evidence appropriate for the Current Point in the Schedule?



# QUASAR Method – Three Phases

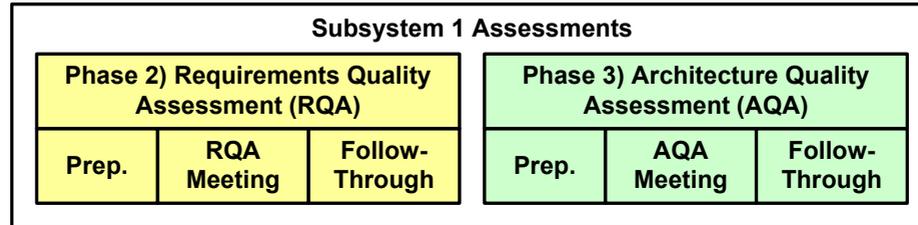
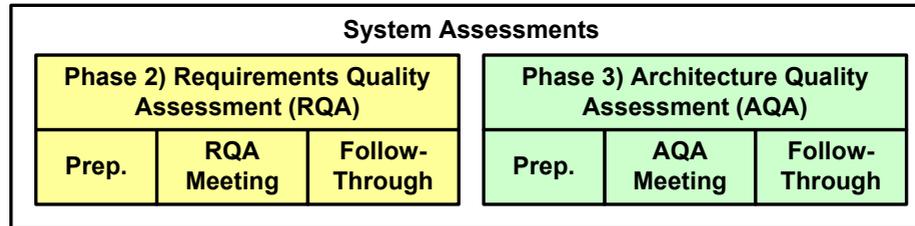
1. Quality Assessment Initiation (QAI)
2. Requirements Quality Assessment (RQA)
3. Architecture Quality Assessment (AQA)



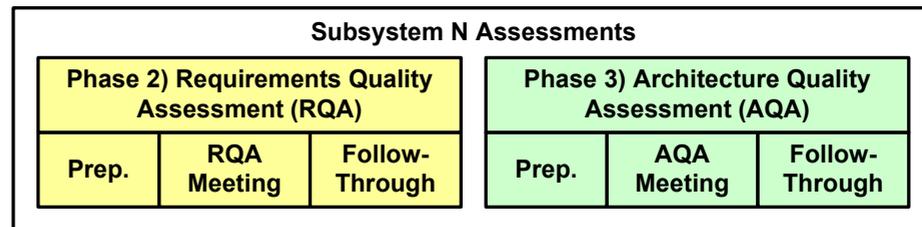
# QUASAR Phases and Tasks



Time (not to scale) →

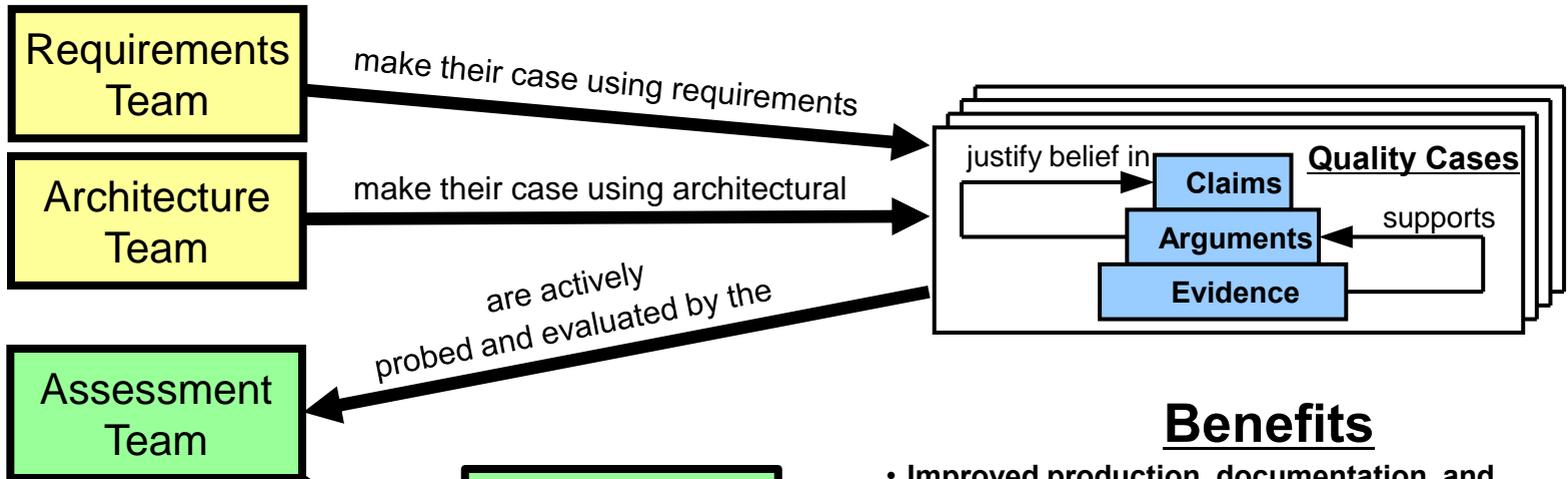


...



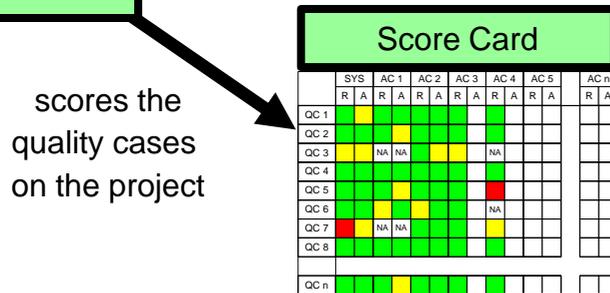
# QUASAR Overview

a method for assessing the *quality*, maturity, and completeness of system architectures and their associated *architecturally significant requirements*

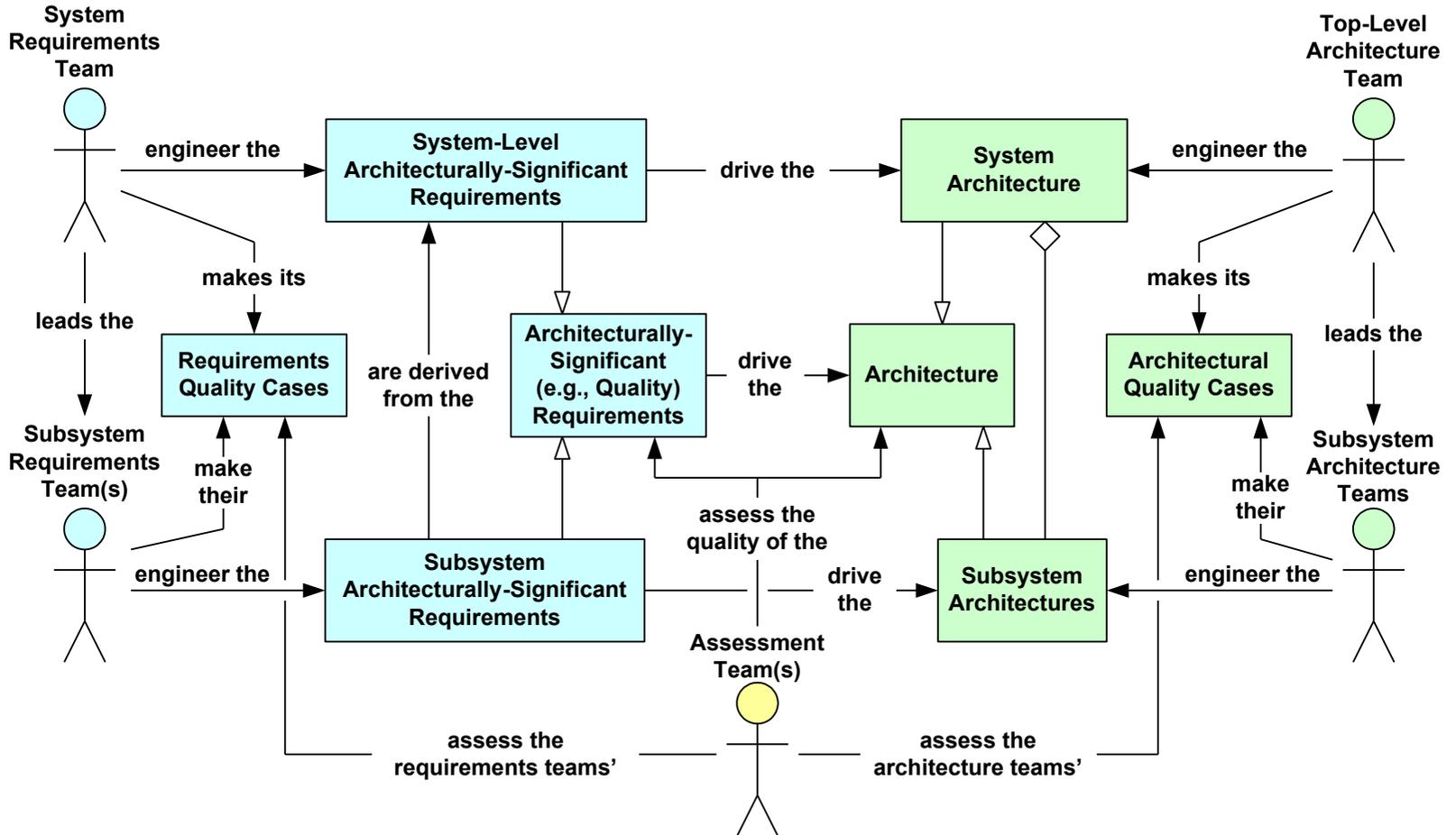


## Benefits

- Improved production, documentation, and verification of architecturally significant requirements, architecture, and architectural representations (quality, maturity, and completeness).
- Improved acquirer visibility into and oversight of the architecturally significant requirements and architecture
- Decreased risks, increased probability of success

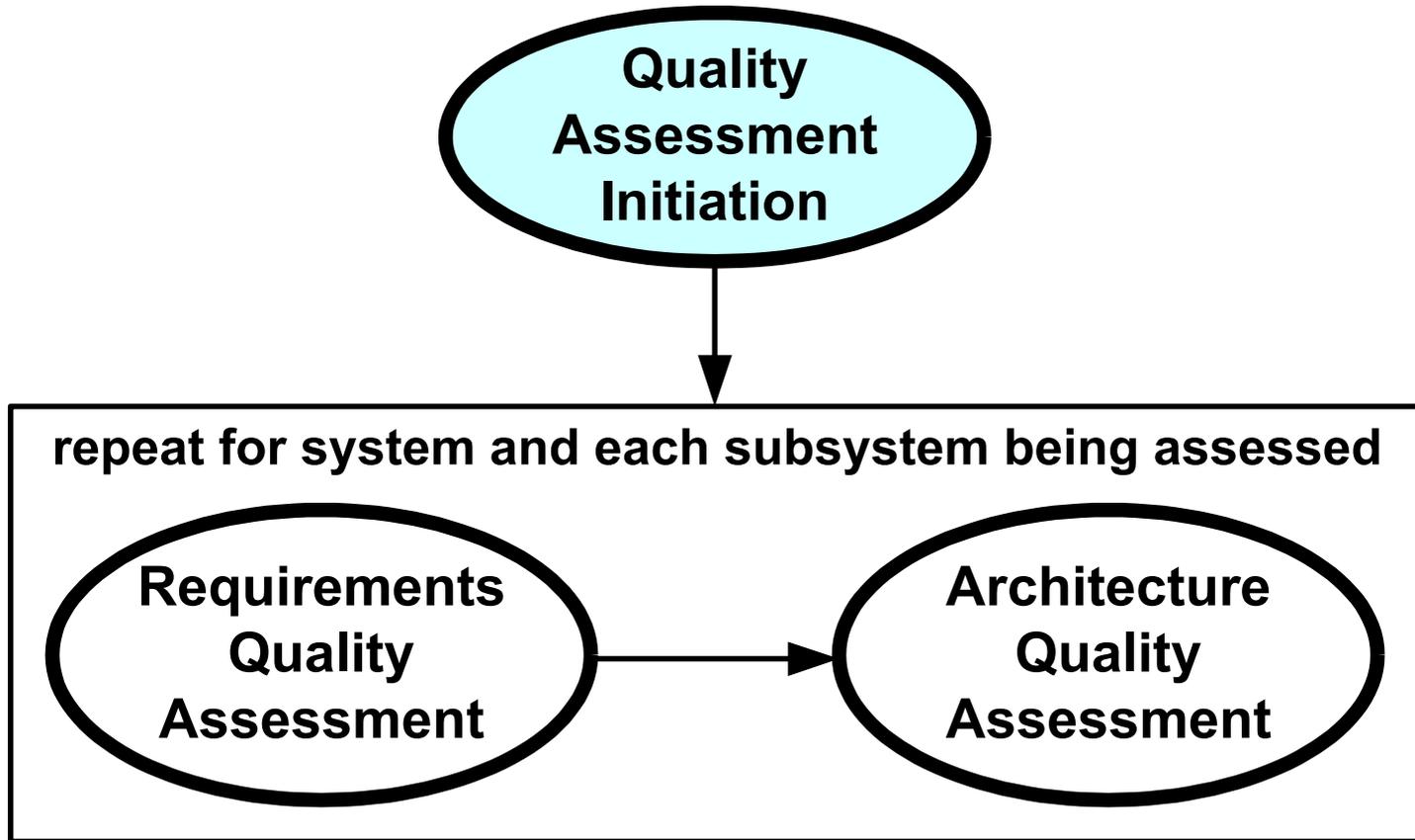


# Quasar Teams and their Work Products



# Quality Assessment Initiation (QAI)

---



# Phase 1) QAI – Objectives

---

Prepare Teams for Requirements and Architecture Assessments

Develop Consensus:

- Scope of Assessments
- Schedule Assessments
- Tailor the Assessment Method and associated Training Materials

Produce and Publish Meeting Outbrief and Minutes

Manage Action Items

Capture Lessons Learned

Tailor/Update QUASAR Method and Training Materials



# Phase 1) QAI – Preparation Task

---

1. Management Team staffs Assessment Team(s)
2. Process and Training Teams train Assessment Team(s)
3. Assessment Team(s) identify:
  - System Requirements Team(s)
  - System Architecture Team(s)
4. Process and Training Teams train System Requirements and Architecture Teams
5. Assessment, Requirements, and Architecture Teams collaborate to Organize QAI Meeting (i.e., Attendees, Time, Location, Agenda)



# Phase 1) QAI – Meeting Task

---

1. Assessment, System Requirements, and System Architecture Teams Collaborate to determine Assessment Scope:
  - Subsystems/Architectural Elements/Focus Areas to Assess (Number and Identity)
  - Quality Characteristics and Quality Attributes underlying Assessment
  - Assessment Resources (e.g., Staffing, Schedule, and Budget)
2. Teams Collaborate to develop Initial Assessment Schedule with regard to System schedule, Subsystem schedule, and associated milestones
3. Teams Collaborate to tailor QUASAR Method
4. Assessment Team captures Action Items



# Phase 1) QAI – Follow-Through Task

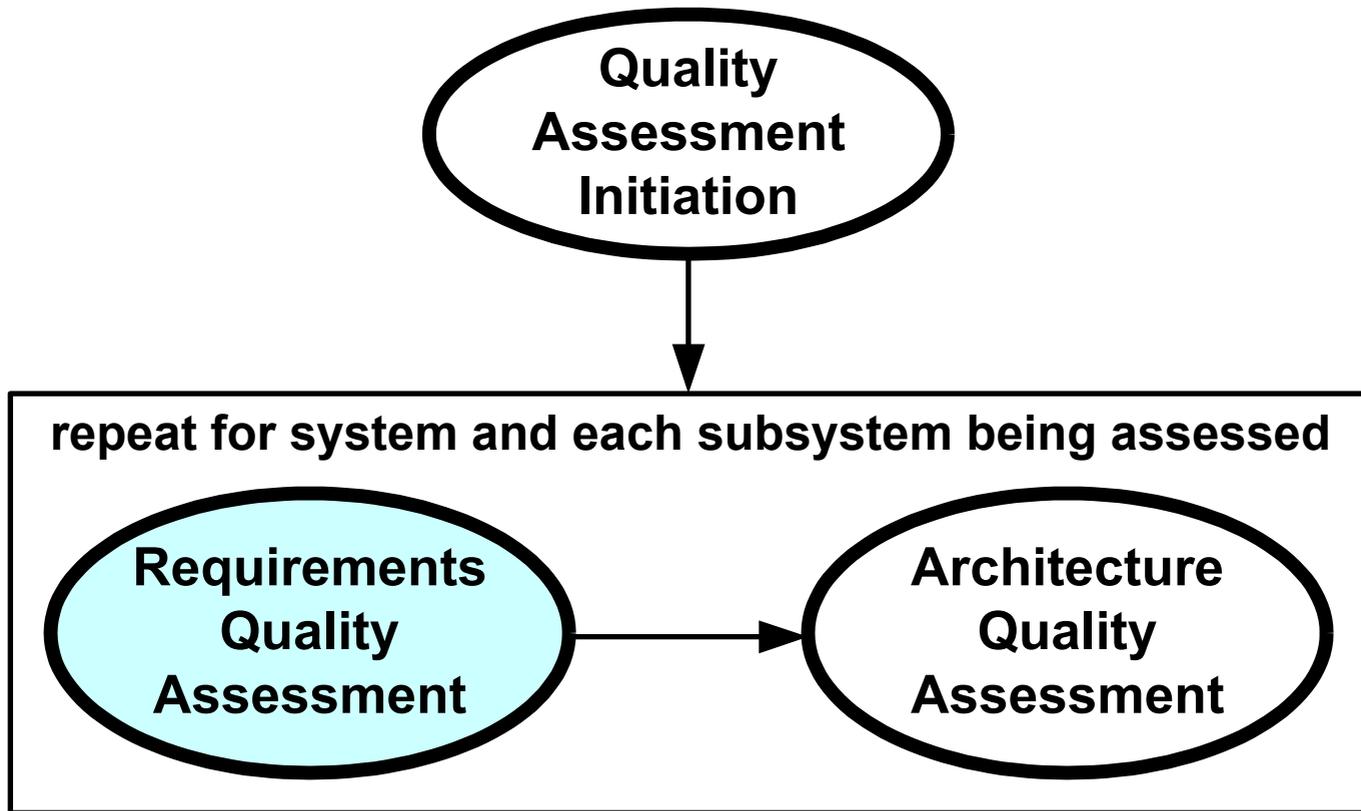
---

1. Assessment Team develops and presents Meeting Outbrief
2. Assessment Team develops, reviews, and distributes Meeting Minutes
3. Assessment/Process/Training Teams tailor, internally review, and distribute:
  - QUASAR Procedure, Standards, and Templates
  - QUASAR Training Materials
4. Teams distribute Assessment Schedule
5. Teams obtain Needed Resources
6. Assessment Team Manages Action Items
7. Assessment Team captures Lessons Learned



# Requirements Quality Assessment (RQA)

---



# Phase 2) ARA – Objectives<sub>1</sub>

---

## Use Requirements Quality Cases to:

- Independently assess Quality and Maturity of the Architecturally Significant Requirements:
  - Drive the Architecture
  - Form Foundation for Architecture Quality Assessment
- Help Requirements Engineers identify Requirements Defects and Weaknesses so that:
  - Defects and Weaknesses can be Corrected
  - The Architecture (and System) can be Improved



# Phase 2) RQA – Objectives<sub>2</sub>

---

Use Requirements Quality Cases to:

- Identify Requirements Risks so that they can be Managed
- Provide Visibility into the Status and Maturity of the Requirements
- Increase the Probability of Project Success

Ensure Architecture Team will be Prepared to Support the coming Architecture Quality Assessment.

Capture Lessons Learned.

Update QUASAR Method and associated Training Materials.



# Phase 2) RQA – Challenges

---

Many Requirements Engineers are not taught how to Engineer Non-functional Requirements including Quality Requirements.

Although popular, Use Case Modeling is not very Effective for Engineering *Quality* Requirements.

Quality Requirements often require the Input from Specialty Engineering Teams (e.g., Reliability, Safety, and Security), who are not often adequately involved during Requirements Engineering.

Quality Goals are often Mistakenly Specified as Quality Requirements.

Architecturally Significant Requirements are typically:

- Incomplete  
(missing important Relevant Quality Characteristics and Attributes)
- Of Poor Quality (lack important characteristics)



# Phase 2) RQA – Preparation Task

---

Process/Training Team trains the Requirements and Architecture Teams *significantly prior* to the RQA Meeting.

Requirements and Architecture Teams provide Preparatory Materials to the Quality Assessment Team *significantly prior* to the RQA Meeting:

- Summary Presentation Materials
- Requirements Quality Cases  
(including electronic access to evidentiary materials)
- Example of Planned Architectural Quality Case

Quality Assessment Team:

- Reads Preparatory Materials
- Generates RFIs and RFAs



# Phase 2) RQA – Meeting Task

---

1. Requirements Team presents:
  - System Overview
  - Requirements Overview
  - *Requirements Quality Cases*
2. Quality Assessment Team assesses Quality and Maturity of Requirements:
  - Completeness of Quality Cases
  - Quality of Quality Cases
3. Architecture Team presents Representative Architectural Quality Case
4. Quality Assessment Team recommends Improvements
5. Quality Assessment Team manages Action Items



# Phase 2) RQA – Follow-Through Task

---

## Quality Assessment Team:

1. Develops Consensus Regarding Requirements Quality
2. Produces, Reviews, and Presents Meeting *Outbrief*
3. Produces, Reviews, and Publishes *RQA Report*
4. Updates and publishes the System Quality Assessment Summary Matrix
5. Captures Lessons Learned
6. Manages Action Items

## Requirements Team:

Addresses Risks Raised in RQA Report

## Process Team:

Updates Assessment Method (e.g., Standards and Procedures)

## Training Team:

Updates Training Materials (if appropriate)



# System Quality Assessment Summary Matrix

	SYS		AC 1		AC 2		AC 3		AC 4		AC 5		AC n	
	R	A	R	A	R	A	R	A	R	A	R	A	R	A
QC 1	Green	Yellow	Green	Green	Green	Green	Green		Green					
QC 2	Green	Green	Green	Yellow	Green	Green	Green		Green					
QC 3	Yellow	Yellow	NA	NA	Green	Yellow	Yellow		NA					
QC 4	Green		Green											
QC 5	Green	Green	Green	Yellow	Green	Green	Green		Red					
QC 6	Green	Green	Yellow	Green	Yellow	Green	Green		NA					
QC 7	Red	Yellow	NA	NA	Green	Green	Green		Yellow					
QC 8	Green		Green											
QC n	Green	Green	Green	Yellow	Green	Green	Green		Green					



# Phase 2) RQA – Checklist

---

Are the Claims:

- Based on the project Quality Model?
- Appropriate for the Current Time in the Project Development Cycle?

Are the Arguments:

- Clear (understandable to the assessors)?
- Compelling (sufficient to justify belief in the claims)?
- Relevant (to justify belief in the claims)?

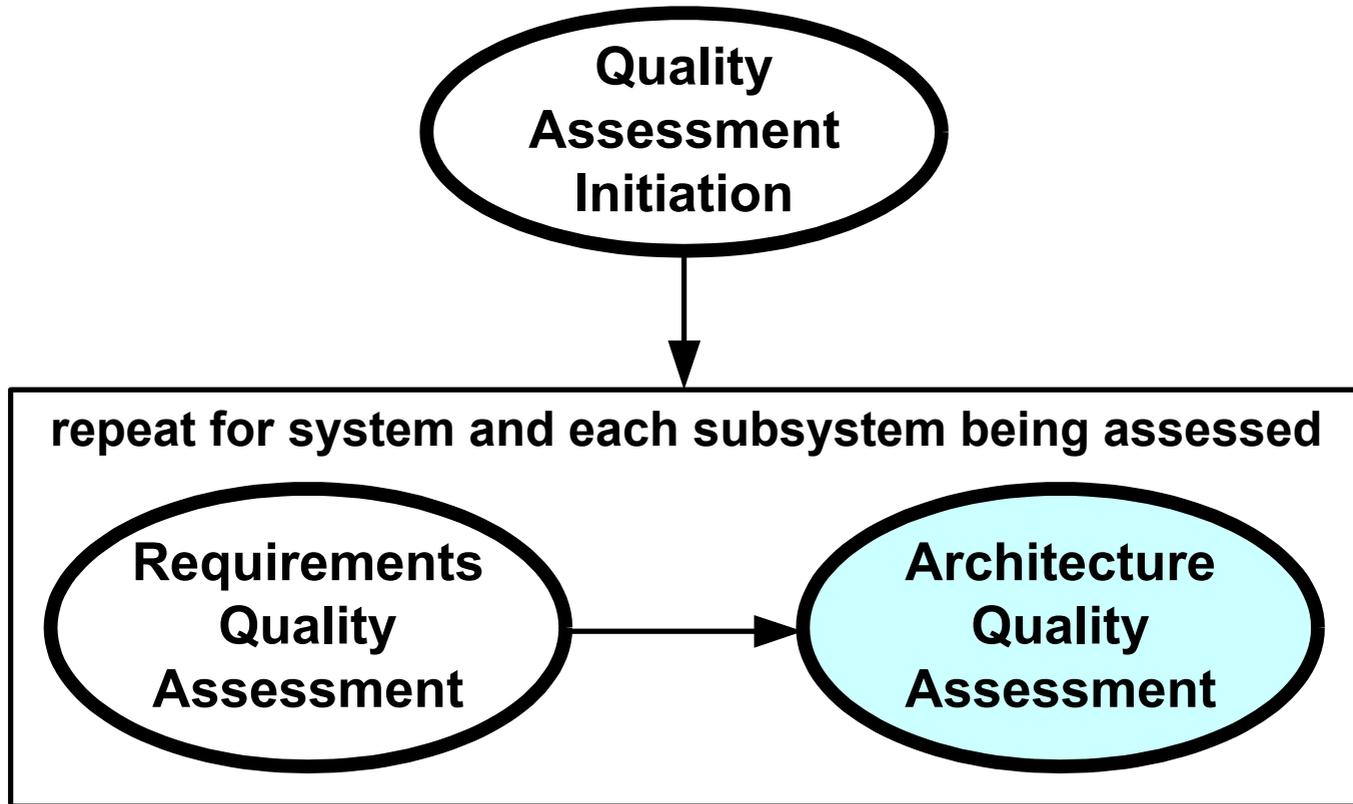
Is the Evidence:

- Credible (official requirements work products under configuration control)?
- Sufficient (to support the arguments)?



# Architecture Quality Assessment (AQA)

---



# Phase 3) AQA – Objectives

---

Use Architectural Quality Cases to:

- Independently assess Architecture Quality in terms of its Support for its Derived and Allocated Architecturally Significant Requirements
- Thereby Verify Compliance with:
  - These Requirements
  - The associated Contract Requirements.
- Help Architects identify Architectural Defects and Weaknesses so that:
  - Defects and Weaknesses can be Corrected
  - The Architecture (and System) can be Improved
- Identify Architectural Risks so that they can be Managed
- Provide Visibility into the Status and Maturity of the Architecture
- Increase the Probability of Project and System Success



# Phase 3) AQA – Principles

---

The Architects should know:

- The Quality Requirements *driving* the Development of the Architecture.
- What Architectural Decisions they *made* and why they made them.
- Where they *documented* their Architectural Decisions.

The Architects should already have documented this Information as a *Natural* Part of their Architecture Engineering Method.

Little *New* Documentation should be Necessary for the Architects to make their Cases to the Quality Assessment Team.

The Architects are Responsible for making their own Cases that their Architecture Sufficiently Supports its Derived and Allocated Quality Requirements.



# Phase 3) AQA – Preparation Task

---

Architecture and Quality Assessment Teams organize the AQA Assessment Meeting.

Training Team provides (at appropriate time):

- QUASAR Training (if not provided prior to RQA assessment)
- AQA Assessment Checklist and Report Template

Architecture Team makes available (min. 2 weeks before meeting):

- Any Updated Quality Requirements
- Architecture Overview
- Quality Case Diagrams
- Architecture Quality Cases (Claims, Arguments, and Evidence)

Quality Assessment Team:

- Reads Preparatory Materials
- Generates RFIs and RFAs



# Phase 3) AQA – Meeting Task

---

## Architecture Team:

1. Introduces the Architecture  
(e.g., Context and Major Functions)
2. Briefly summarizes the Architecturally Significant Requirements
3. Briefly summarizes the Architecture  
(e.g., Most Important Architectural Components, Relationships, Decisions, Inventions, Trade-Offs, Assumptions, and Rationales)
4. Individually Presents Architectural Quality Cases  
(Quality Case Diagram, Claims, Arguments, and Evidence)

## Quality Assessment Team:

1. Probes Architecture (Architectural Quality Case by Quality Case)
2. Manages Action Items



# Phase 3) AQA – Follow-Through Task

---

## Quality Assessment Team:

1. Develops Consensus regarding Architecture Quality
2. Produces, reviews, and presents Meeting Outbrief
3. Produces, reviews, and publishes AQA Report
4. Updates and republishes System Quality Assessment Summary Matrix
5. Captures Lessons Learned
6. Manages Action Items

## Architecture Team:

Addresses Architectural Defects, Weaknesses, and Risks Raised in AQA Report

## Process Team:

Updates Assessment Method (if appropriate)

## Training Team:

Updates Training Materials (if appropriate)



# Phase 3) AQA – Checklist

---

Are the Claims:

- Based on the project Quality Model?
- Appropriate for the Current Time in the Project Development Cycle?

Are the Arguments:

- Clear (understandable to the assessors)?
- Compelling (sufficient to justify belief in the claims)?
- Relevant (to justify belief in the claims)?

Is the Evidence:

- Credible (official architecture work products under configuration control)?
- Sufficient (to support the arguments)?



# Phase 3) AQA – Team Memberships

---

## Quality Assessment Team (Assessors):

- Assessment Team Leader
- Meeting Facilitator
- Acquirer/Customer Liaisons to Developer:
  - Requirements Teams
  - Architecture Teams
- Subject Matter Experts (SMEs) having adequate training and experience in:
  - Application Domains  
(e.g., avionics, sensors, telecommunications, and weapons)
  - Specialty Engineering Groups  
(e.g., reliability, safety, and security)
  - Requirements and Architecture Engineering (including Quality Model)
  - QUASAR
- Scribe
- Acquirer/Customer *Observers*



# Key Lessons Learned

---

Quality Cases are a very effective and efficient way to assess existence of and compliance with architecturally significant requirements.

Include architectural quality (requirements and architecture) assessments in the contract so that they can be budgeted, scheduled, staffed, and enforced.

It is better to organize the assessments by quality characteristics than subsystems.

ATAMs and QUASARs are both useful, having different strengths and “sweet-spots”:

- ATAMs:
  - Software Architecture
  - Architecture Improvement
- QUASARs:
  - System Architecture
  - Requirements and Requirements Compliance



This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

## NO WARRANTY

THIS MATERIAL OF CARNEGIE MELLON UNIVERSITY AND ITS SOFTWARE ENGINEERING INSTITUTE IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

