Introduction to Systems Architecture

Lecture 01, v01

John G. Artus

BSEE

MSSE

INCOSE ESEP

About This Courseware

- The majority of the material presented in this course is sourced from the textbook “System Architecture” by Edward Crawley, Bruce Cameron, and Daniel Selva
- I, John Artus, make no claim of ownership of the material sourced from this textbook
- I, John Artus, am using the material sourced from this textbook, and other indicated sources, as content for this courseware for educational purposes only
- This courseware lecture material has been sourced, interpreted, assembled, formatted, and copyrighted by John G. Artus for use in this educational context
- Anyone may freely access, and reuse this material in an educational context provided the copyright owner, John G. Artus, is recognized as the interpreter, assembler, and formatter of the source material used in the generation of this courseware, and provided that Edward Crawley, Bruce Cameron, and Daniel Selva are recognized as authors of the textbook “System Architecture” from which the majority of the content of this courseware has been sourced

Preface

- This course is about Systems Architecture
- There are many kinds of systems
  - Natural
  - Social
  - Mechanical
  - Electrical
  - Biological
  - Informational
  - Software
  - etc

- While the course textbook “Systems Architecture” by Crawley, Cameron, and Selva includes discussions of various types of systems, the focus in this course will be on Complex Engineered Systems
  - Simple system examples will be used to explore the processes and practices used to analyze, design, and develop more complex systems
- Engineered Systems are ethical socio-technical systems with a recognized SE life cycle
  - They are engineered by humans, utilizing technology developed by humans, to provide a socially-beneficial service to humans
  - Natural and social systems are considered only as far as their relevant and important environmental considerations pertain to the engineered System of Interest (SoI)

What is a "System"

- ISO/IEC/IEEE 15288 – Systems Engineering Processes
  - Systems are man-made, created and utilized to provide products and services in defined environments for the benefit of users and other stakeholders
  - A system is an arrangement of parts or elements that together exhibit behavior or meaning that the individual constituents do not
- SEBoK – SE Body of Knowledge
  - A system is any set of related parts for which there is sufficient coherence between the parts to make viewing them as a whole useful

It is VERY difficult to summarize all there is to say about a System in one statement (no matter how carefully crafted it is)

- Main points
  - A system is made up of entities that interact or are interrelated
  - Relationships among entities can be
    - Static – as with a connection
    - Dynamic – as with an exchange of items
      - Information
      - Material
      - People
      - Energy
      - Data
  - Key takeaway:
    - A system is a set of entities and their relationships, whose *functionality is greater than that of the sum of the individual entities*

Entities are the parts, modules, routines, assemblies, etc, that make up the whole

In this course we are primarily interested in *engineered systems* – those conceived of and developed by human beings

A Note About Software Systems

- Software Systems are systems in which the vast majority of the system elements are software components
- Software is a recognized element type of complex engineered systems
- Complex engineered systems can involve a large number of software components
- Software Engineering is a specialized domain of Systems Engineering
  - The processes and procedures are specialized forms of the general Systems Engineering processes and procedures
  - Specifically designed to deal with the design, development, and integration of software components
- While all of the lessons in this course apply equally to software components, the focus of the lecture material will be on traditional components that may or may not include software components
- The detailed processes and procedures for dealing specifically with software components is left to a be covered in a course on Software Engineering

What is “Systems Engineering”

- INCOSE
  - Systems Engineering is a transdisciplinary and integrative approach to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods.
- ISO/IEC/IEEE 15288 and NASA Systems Engineering Handbook
  - Interdisciplinary approach governing the total technical and managerial effort required to transform a set of customer needs, expectations, and constraints into a solution and to support that solution throughout its life.

It is VERY difficult to summarize all there is to say about Systems Engineering in one statement (no matter how carefully crafted it is)

- Good Systems Engineering work results in a system that displays these characteristics:
  - The system is easily identifiable, including its boundary, its form, and its function
  - The entities that make up the system are easily identified, including their boundary, their form, and their function
  - The relationships among the entities that make up the system are easily identified, including their boundary, their form, and their function; this includes the relationships that cross the system boundary (to other external systems)
  - The emergent properties of the system are identified based on the function of the entities, and their functional interactions

What is a “System Architecture”

- ISO/IEC/IEEE 15288 – System Life Cycle Processes
  - A system architecture represents the fundamental concepts or properties related to a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution
- SEBoK – SE Body of Knowledge
  - The solution architecture has features, properties, and characteristics which satisfy, as far as possible, the problem or opportunity expressed by a set of system requirements (traceable to mission/business and stakeholder requirements) and life cycle concepts (e.g., operational, support) and which are implementable through technologies (e.g., mechanics, electronics, hydraulics, software, services, procedures, human activity).
- ISO/IEC/IEEE 42010 – Architecture Description
  - A system architecture establishes the fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution

It is VERY difficult to summarize all there is to say about a System Architecture in one statement (no matter how carefully crafted it is)

- In summary
  - A System Architecture is an abstract description of the entities of a system and the relationships between those entities
  - A good System Architecture displays the following characteristics
    - Meets stakeholders’ needs and delivers value
    - Integrates easily with other entities (other systems)
    - Evolves flexibly to meet changing requirements over time
    - Operates simply and reliably

What is “System Architecting”?

- ISO/IEC/IEEE 15288 – System Life Cycle Processes
  - The purpose of the Architecture Definition process is to generate system architecture alternatives
  - To select one or more alternative(s) that frame stakeholder concerns and meets system requirements
  - To express the Architectural Description in a set of consistent views
- SEBoK – SE Body of Knowledge
  - The purpose of system architecture activities is to define a comprehensive solution based on principles, concepts, and properties logically related to and consistent with each other
- ISO/IEC/IEEE 42010 – Architecture Description
  - The process of architecting is that of conceiving, defining, expressing, documenting, communicating, certifying proper implementation of, maintaining, and improving an architecture throughout a system’s life cycle.

It is VERY difficult to summarize all there is to say about System Architecting in one statement (no matter how carefully crafted it is)

- The key purposes of performing the System Architecting process:
  - Ensure that the solution, as described in the architecture
    - Achieves the required performance and resource levels (a set of defined targets and constraints)
    - Is distributed to and available to the entire development team
  - Provide the architectural detail needed by the design process engineers to continue the system development

What is a “System Architectural Description”?

- ISO/IEC/IEEE 42010 – Architecture Description
  - The Architectural Description is a work product used to express an architecture
    - It includes one or more architecture views
    - An architecture view addresses one or more of the concerns held by the system’s stakeholders
    - An architecture view expresses the architecture of the system-of-interest in accordance with an architecture viewpoint
    - There are two aspects to a viewpoint: the concerns it frames for stakeholders and the conventions it establishes on views

There is no single statement from accepted standard documentation that concisely and adequately defines what is an Architectural Description

- From the following documents, an short, but adequate definition of an Architectural Description can be assembled
  - ISO/IEC/IEEE 15288 – System Life Cycle Processes
  - ISO/IEC/IEEE 42010 – Architecture Description
  - SEBoK – SE Body of Knowledge
  - INCOSE Handbook v4
- An Architectural Description is the expression of an architecture in the form of Architectural Views which address the concerns held by the system’s stakeholders
- Architectural Viewpoints establish the conventions by which Architectural Views are presented in an Architectural Description

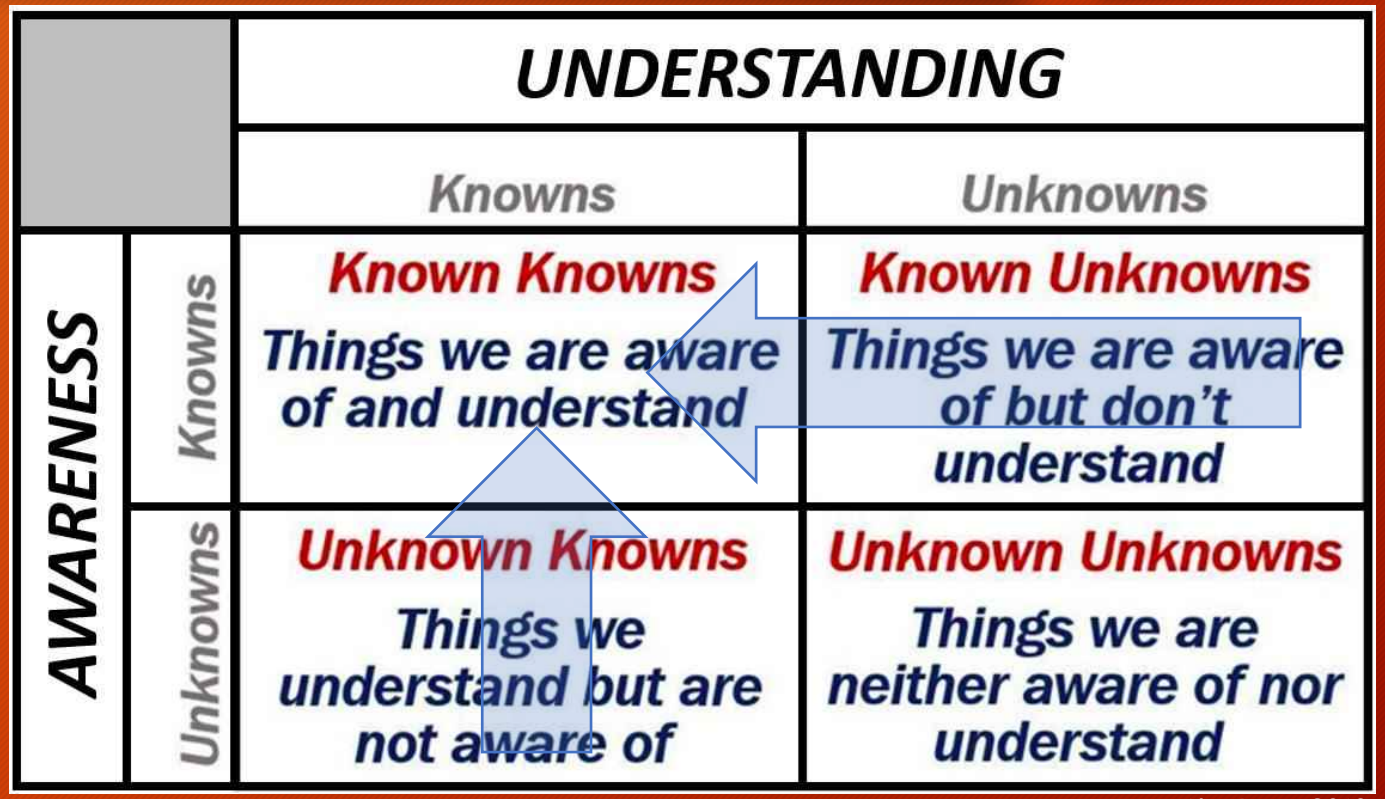
Examples of Stakeholder Concerns

- Affordability
- Agility
- Alignment With Business Goals & Strategies
- Assurance
- Autonomy
- Availability
- Behavior
- Business Impact
- Capability
- Complexity
- Compliance To Regulation
- Concurrency
- Control
- Cost
- Customer Experience
- Data Accessibility
- Deadlock
- Disposability
- Environmental Impact
- Feasibility
- Flexibility
- Functionality
- Information Assurance
- Interoperability
- Inter-process Communication
- Known Limitations
- Maintainability
- Misuse
- Modifiability
- Modularity
- Openness
- Performance
- Privacy
- Quality Of Service
- Reliability
- Resilience
- Resource Utilization
- Schedule
- Security
- Shortcomings
- State Change
- Structure
- Subsystem Integration
- System Features
- System Properties
- System Purposes
- Usability
- Usage
- Viability

Source: Overview of an Emerging Standard on Architecture Evaluation – ISO/IEC 42030; James Martin; 2017; INCOSE

Rumsfeld's "Known Unknowns" (etc) *

- One goal of System Architecture work is to identify those areas lacking either understanding or awareness
- Especially those having a major impact on stakeholder value
- Need to resolve them (as able) to become Known Knowns
  - This occurs as the architecture is further developed at lower levels of detail
- Unknown Unknowns are particularly dangerous
  - Discovering too many of these may mean you are in the wrong business



Ahn Dang, 2019

While these concepts were new and interesting to the public in 2002, Rumsfeld did not "discover" them. These concepts were well known to and understood by Systems Engineers for decades prior.

Systems Engineers translate "Lack of Awareness and Understanding" into "Risk" and "Uncertainty"

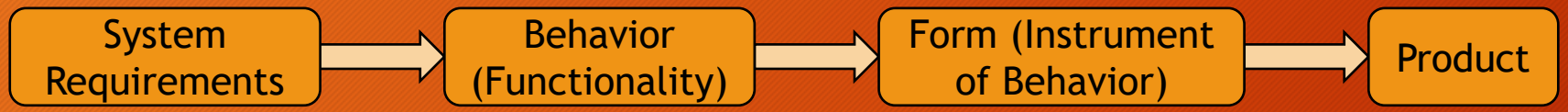
* **Donald Rumsfeld (February 12, 2002).** *United States Department of Defense*

Emergence

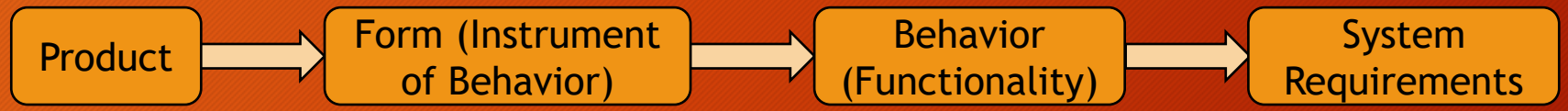
- Emergence refers to what appears, materializes, or surfaces when a system operates
  - Understanding emergence is a goal of system thinking
- Function is what a system does: its actions, outcomes, or outputs
  - Function emerges when a system components are integrated together
- Humans design systems so that the anticipated desirable primary function may emerge
  - This primary function is often linked to the benefit produced by the system
  - Often, other than the primary desirable function can emerge from a system
- Emergent function can be anticipated or unanticipated, and it can be desirable or undesirable
- Sometimes, as a system comes together, unanticipated function emerges
  - This can either be desirable unanticipated outcome or undesirable unanticipated outcome
- Anticipated but undesirable outcomes may also emerge
  - Architects work to reduce the possibility of emergence of anticipated but undesirable outcomes
  - An excessive amount of “unknowns” in a system architecture increases the risk of anticipated but undesirable outcomes emerging

Forward Engineering versus Reverse Engineering

- In Forward Engineering, conceptualize and specify a problem needing a solution in the form of requirements, that are then developed as behaviors that the system must perform though the implementation of structure or form
- The Forward Engineering process is summarized as:
  - Use the System Requirements to design and build the system
  - Design the Behavior that satisfies the System Requirements
  - Design the Form that instruments the needed Behavior



- In Reverse Engineering, employ deductive reasoning in order to understand 1) what a system does, 2) how the system accomplishes what it does, 3) what was the original system specification
- The Reverse Engineering process is summarized as:
  - Examine the Form of the system, in order to
  - Deduce the Behavior of the system, so as to then
  - Deduce the System Requirements that were originally established to design and build the system



https://www.istockphoto.com/photo/engineer-overseeing-automated-production-process-in-a-factory-gm1152220713-312526676

https://www.nasa.gov/centers/johnson/engineering/human_space_vehicle_systems/energy_systems_test_area/pyrotechnics/index.html

Use of System Analysis in Architecture Work

- Types of system analysis
  - Top-Down
    - Start with a system boundary and an overall description of system functions
    - Through the repeated application of element identification, division, grouping, and allocation of functions, a complete description of the elements needed for the SoI can be defined
    - The choice of system elements and allocation of functions may be guided by
      - Pre-defined ways of solving a given problem
      - Or by identified system patterns
    - Both can support as well as insert bias into the synthesis
  - Bottom-Up
    - Start with major elements and interactions
    - Use division, grouping, and identification to construct a full system description
      - Capable of providing all the necessary functions
      - At which point the final SoI boundary can be set
    - The choice of system elements and groupings will be driven by the goal of ensuring that the major system elements can be formed together into a viable system whole
  - In-Out
    - A combination of the two above which starts with the “knowns”
    - Expand from the knowns to discover and develop the “unknowns”

Forward Engineering often relies heavily on Top-Down analysis

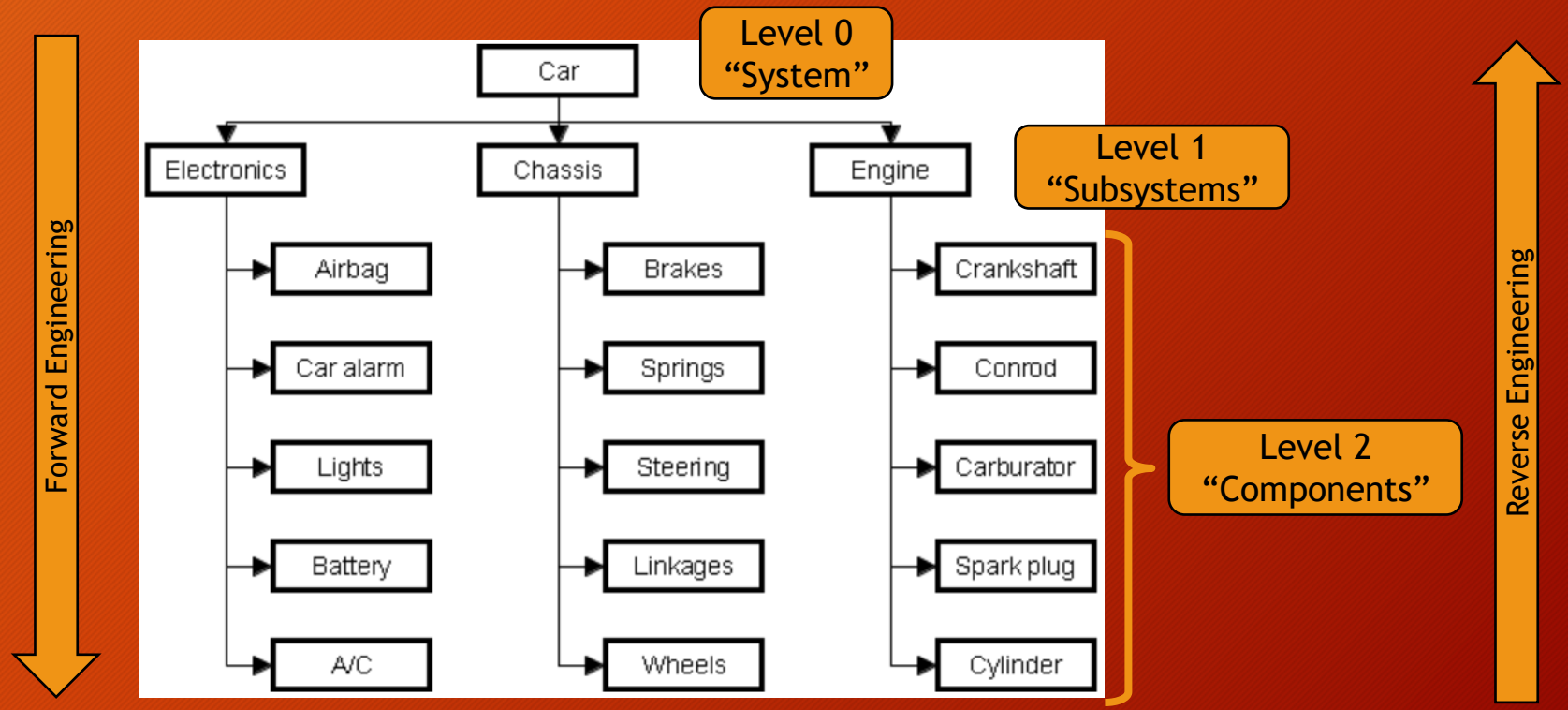
Reverse Engineering often relies heavily on Bottom-Up analysis

Use of Analysis Types in Practical Work

- Different jobs call for different applications of analysis types, often based on experience
  - Experts in the Field
    - Companies with established product lines draw on a significant bank of experience and sets of existing designs to determine the viability of and configuration of future products
    - In situations like this, engineers rely on
      - Bottom-Up analysis to use existing designs to satisfy new requirements
      - Top-Down analysis, where needed to explore new requirements not faced before
  - Second-Class Competitor
    - Companies with less-established product lines must lean more heavily on Top-Down analysis to deal with a significant number of “unknowns”
  - New Entrant into Industry
    - May have to perform a significant amount of Reverse Engineering, employing Bottom-Up analysis of competitor products to gain enough knowledge to become competitive
    - Once sufficient knowledge has been gained, Top-Down analysis of customer requirements will be needed to help get the company on level playing field with the competition

System Hierarchy

- Simplified Automobile Example
  - Car is broken down into its major subsystems
  - Each of the major subsystems is in turn (recursively) broken down into it's major components
  - The process continues recursively until we've either
    - Reached elements which can no further be decomposed, or
    - Have achieved the objective of the decomposition and no further decomposition is needed



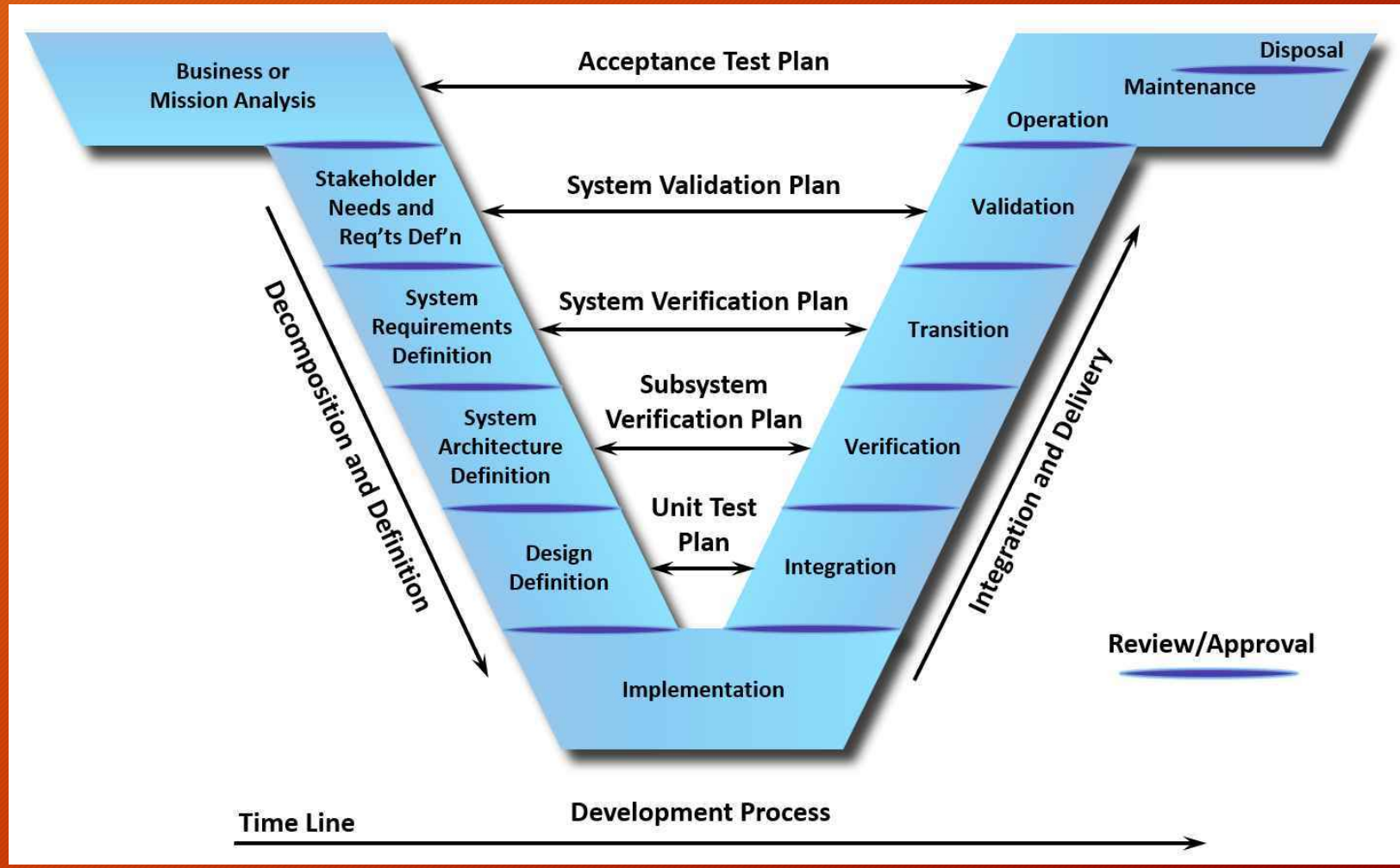
https://whatsoftwarecando.org/en/juggling-trees-layout-hierarchical-information-compactly/

Forward and reverse engineering are not strictly performed in these directions
There is a bidirectional nature to both
But basically, the majority of the work is in the directions indicated

When discussing Systems Engineering proceses, the context is Forward Engineering

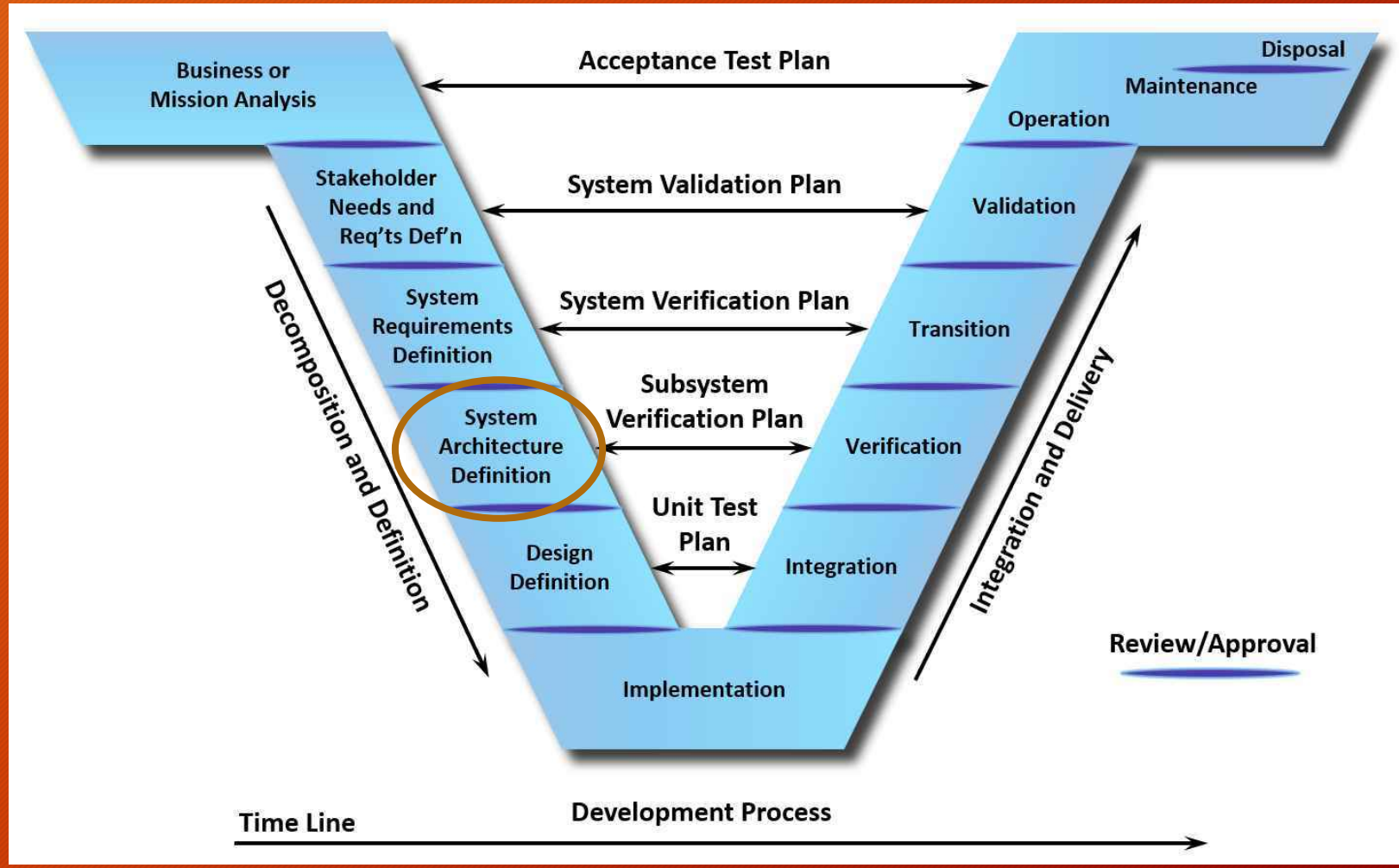
ISO 15288 Systems Engineering "VEE" Model

- The Systems Engineering Lifecycle Processes defined in ISO/IEC/IEEE 15288 can be laid out in a "VEE" format
- This format emphasizes the decomposition and development of the system element details on the left side
- Along with the product integration and delivery details on the right side
- The model is in the shape of a "VEE" to emphasize the parallel relationship between the left side (decomposition and development) with the right side (integration and delivery)
- Verification and Validation planning are done on the left
- Verification and Validation implementation are done on the right



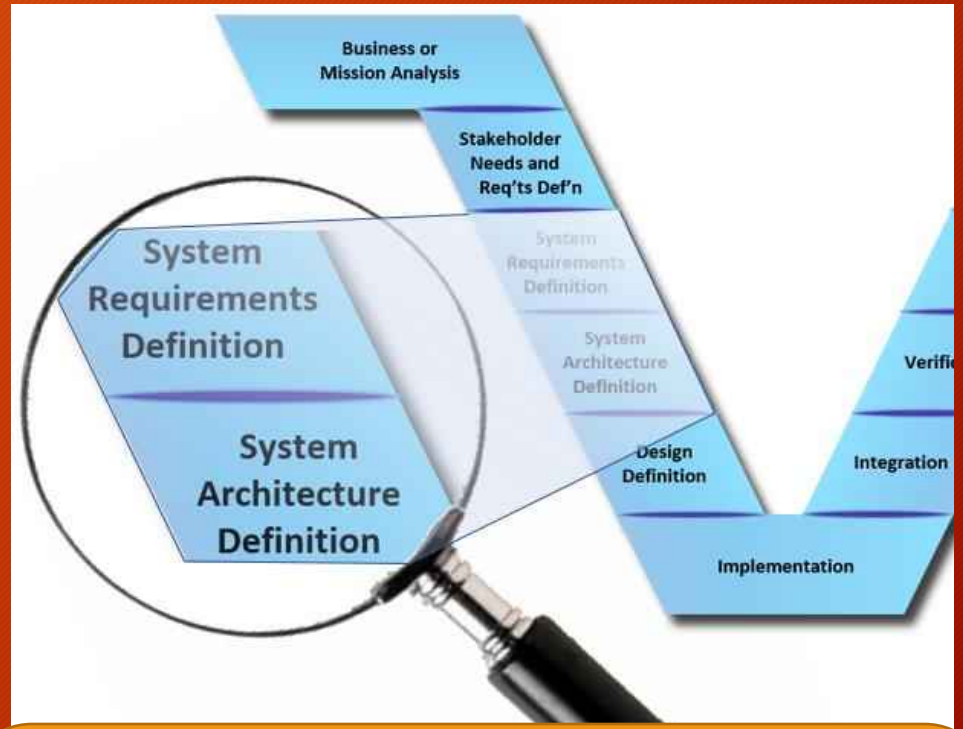
Where Does System Architecting Fit?

- … in the System Life Cycle Process Model VEE?
- System Requirements are input to the System Architecture Definition Process
- The Architecture Description is output to the Design Definition Process



Where Do Requirements End and Architecture Begin?

- The Systems Engineering Life Cycle Processes are iterative and recursive
- Therefore they occur repeatedly, until a review or other program-level decision is made to terminate a process
- During an iteration, however, the System Requirements Definition Process ends when:
  - A preplanned stopping point is reached
    - Due to budget, schedule, or other constraint or milestone
  - Sufficient risk and/or uncertainty has been resolved
  - Architecturally-relevant requirements can no longer be discovered
  - Other program-level decision terminates the process
- However, the complete set of artifacts needed by the Architecture Definition team must be made available prior to termination of the System Requirements Definition Process
- STOP developing requirements when sufficient system specification is provided to the System Architecture Definition Process



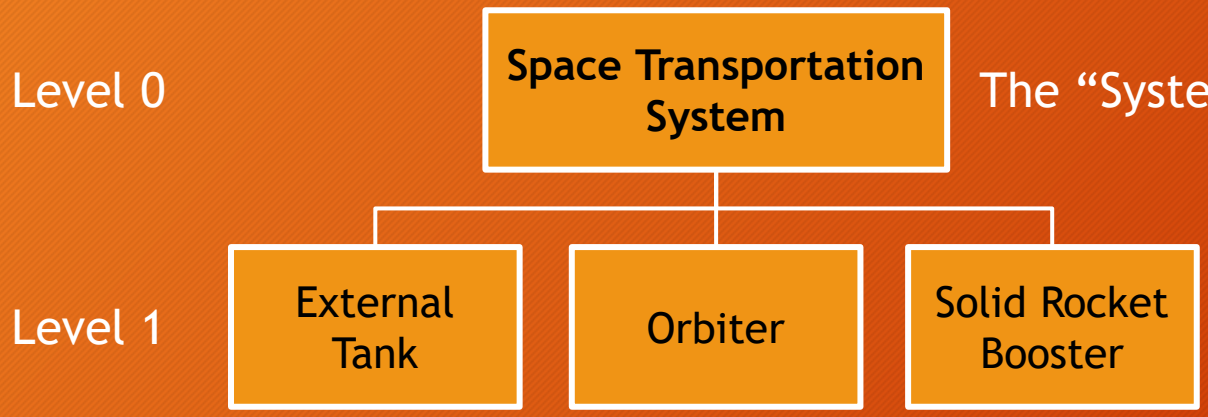
- Specification of System Requirements does not occur in one fell swoop
- Requirements can be fed to the Architecture Design process in stages (by levels)
- Process iteration allows feedback between Architecture and Requirements to further refine the requirements
- However, the Architecture Development process is not complete until all requirements have been considered

System Decomposition Terminology

- There are many different interpretations of the terminology to be used to describe the items which result from a system decomposition
- NASA
  - Element of a system – any item resulting from the system decomposition process
    - Can include hardware, software, people, facilities, policies, documents, databases, etc
    - Any "thing" that is relevant to the system design and operation
  - System - an integrated set of elements that accomplish a defined objective
  - Subsystem- a system in its own right, except it normally will not provide a useful function on its own, it must be integrated with other subsystems (or systems) to make up a system
  - Component - the elements that make up a subsystem or system
  - Part – the elements on the lowest level of the hierarchy

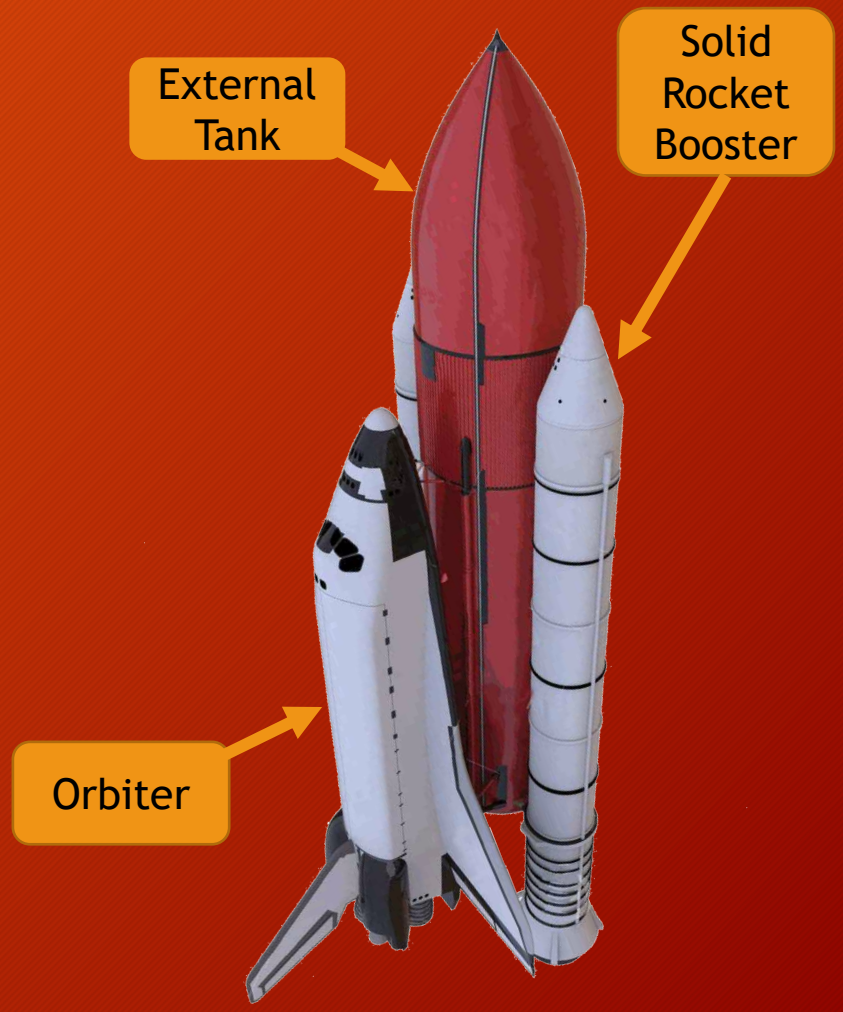
Different organizations use their own set of terms to describe items in a system decomposition
There is no standard utilization of terms across industry at this time

Decomposing the Space Transportation System



It is easier to identify element boundaries when the elements have a clear separation of physical form as in this example

In more difficult cases of decomposition, logical features, such as functionality, may dictate where element boundaries are drawn



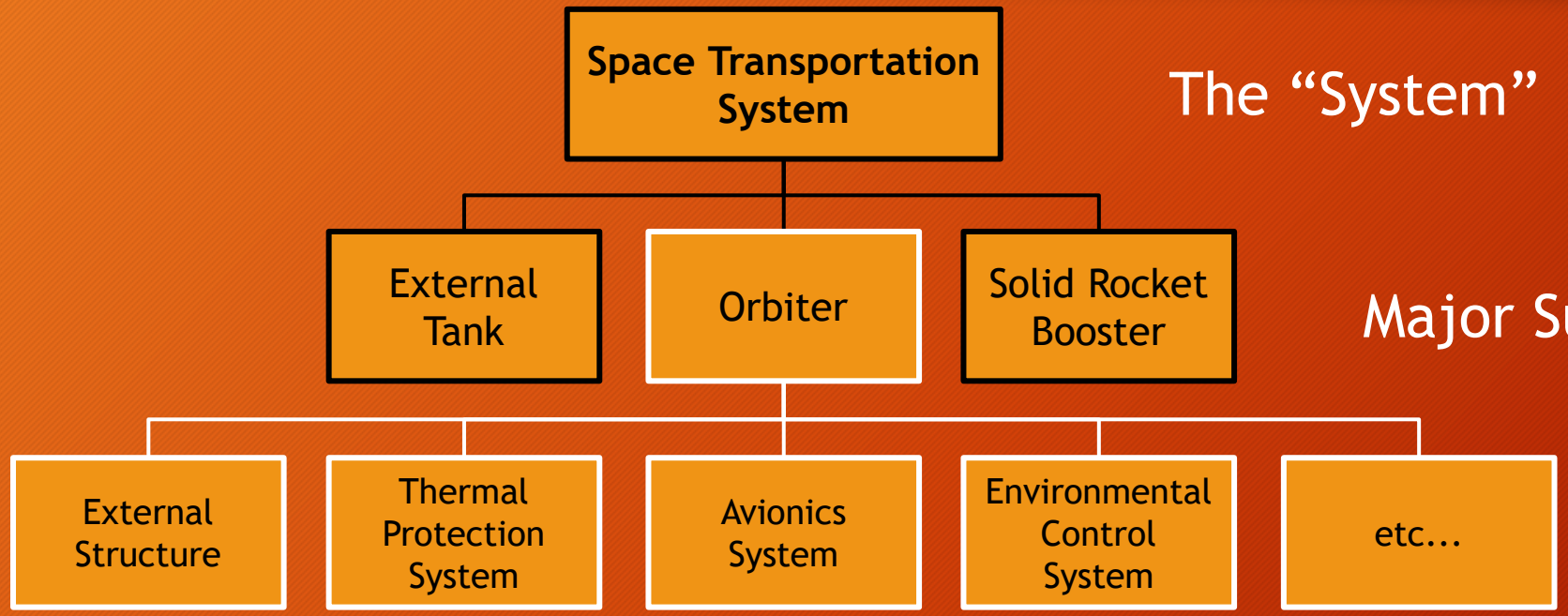
https://thumbs.dreamstime.com/z/space-shuttle-illustration-d-rendered-object-isolated-white-background-no-shadows-70154189.jpg

Decomposing the STS Orbiter

Level 0

Level 1

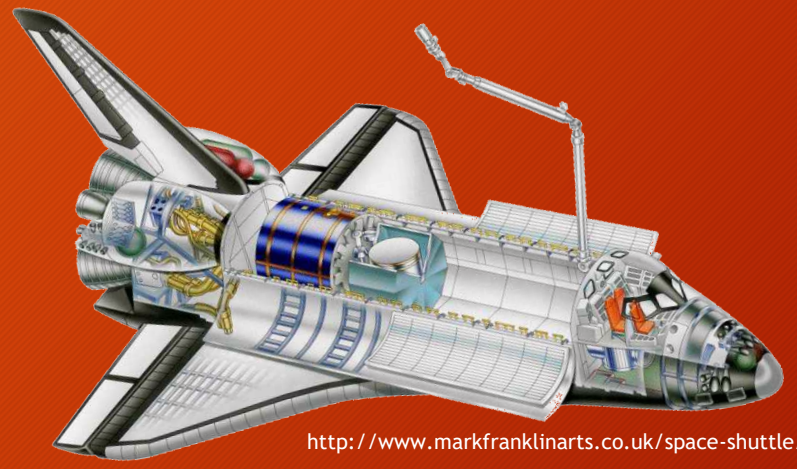
Level 2



The “System”

Major Subsystems

Major Assemblies



http://www.markfranklinarts.co.uk/space-shuttle/

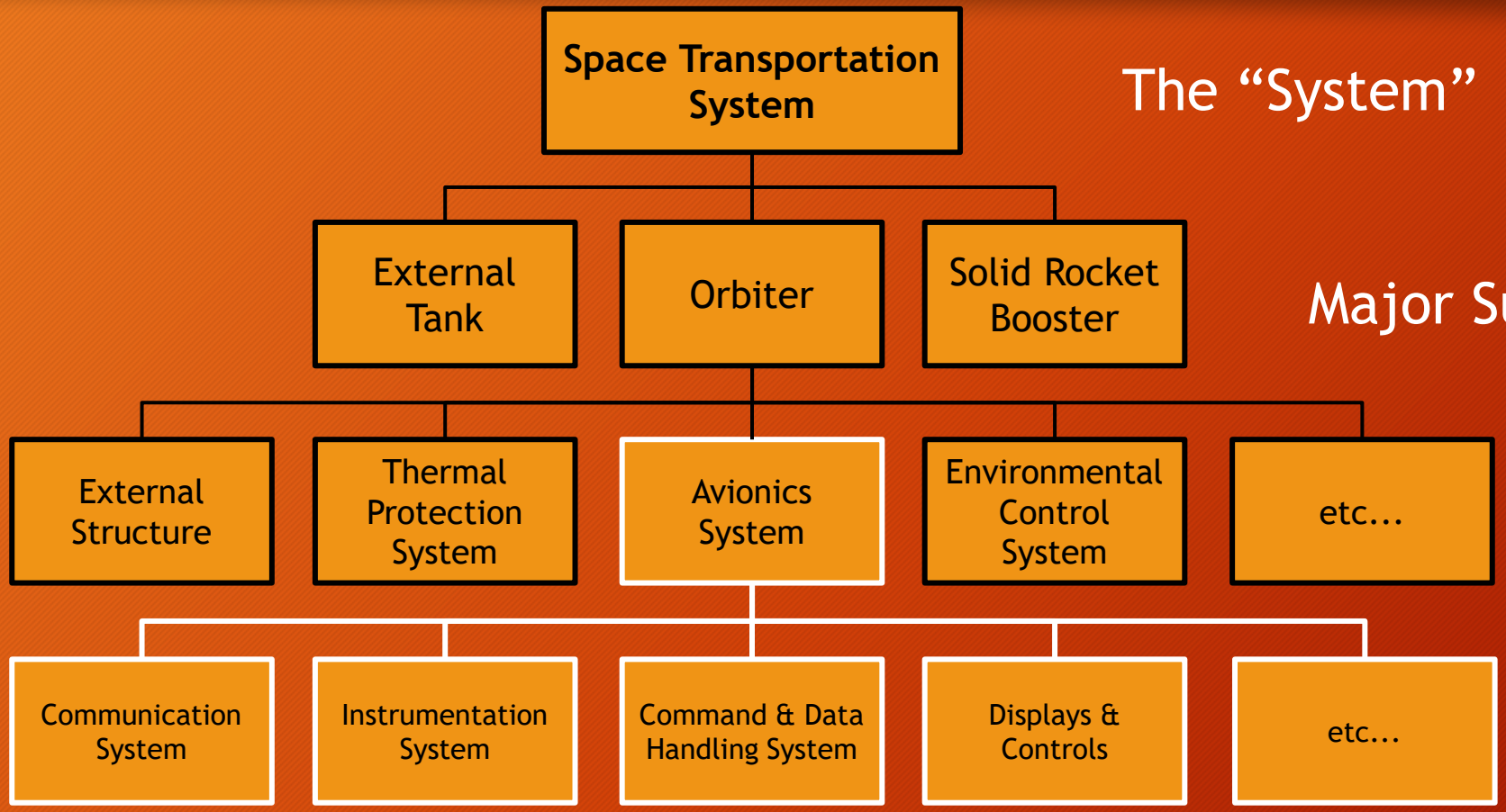
Decomposing the STS Orbiter Avionics System

Level 0

Level 1

Level 2

Level 3

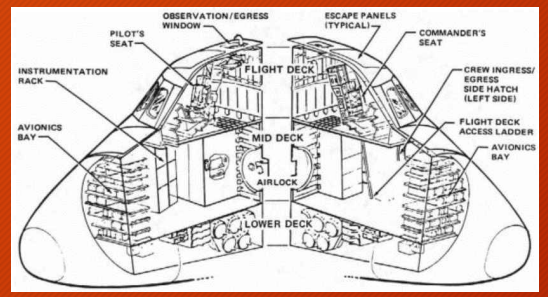


The "System"

Major Subsystems

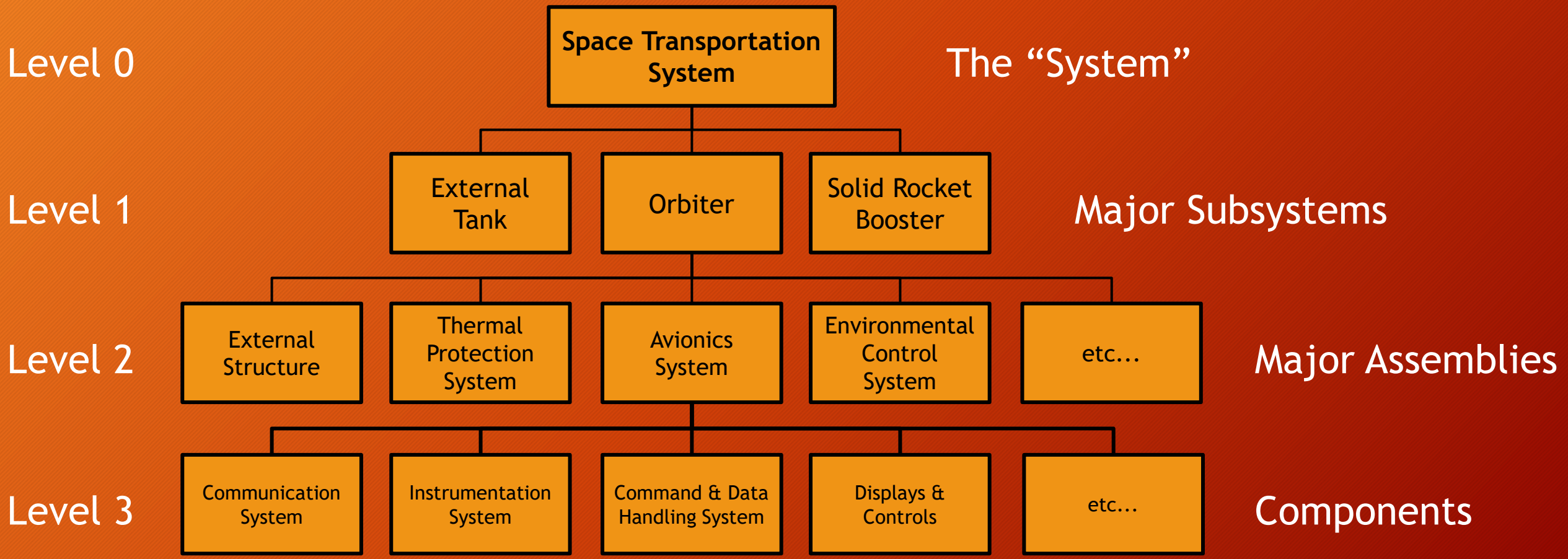
Major Assemblies

Components



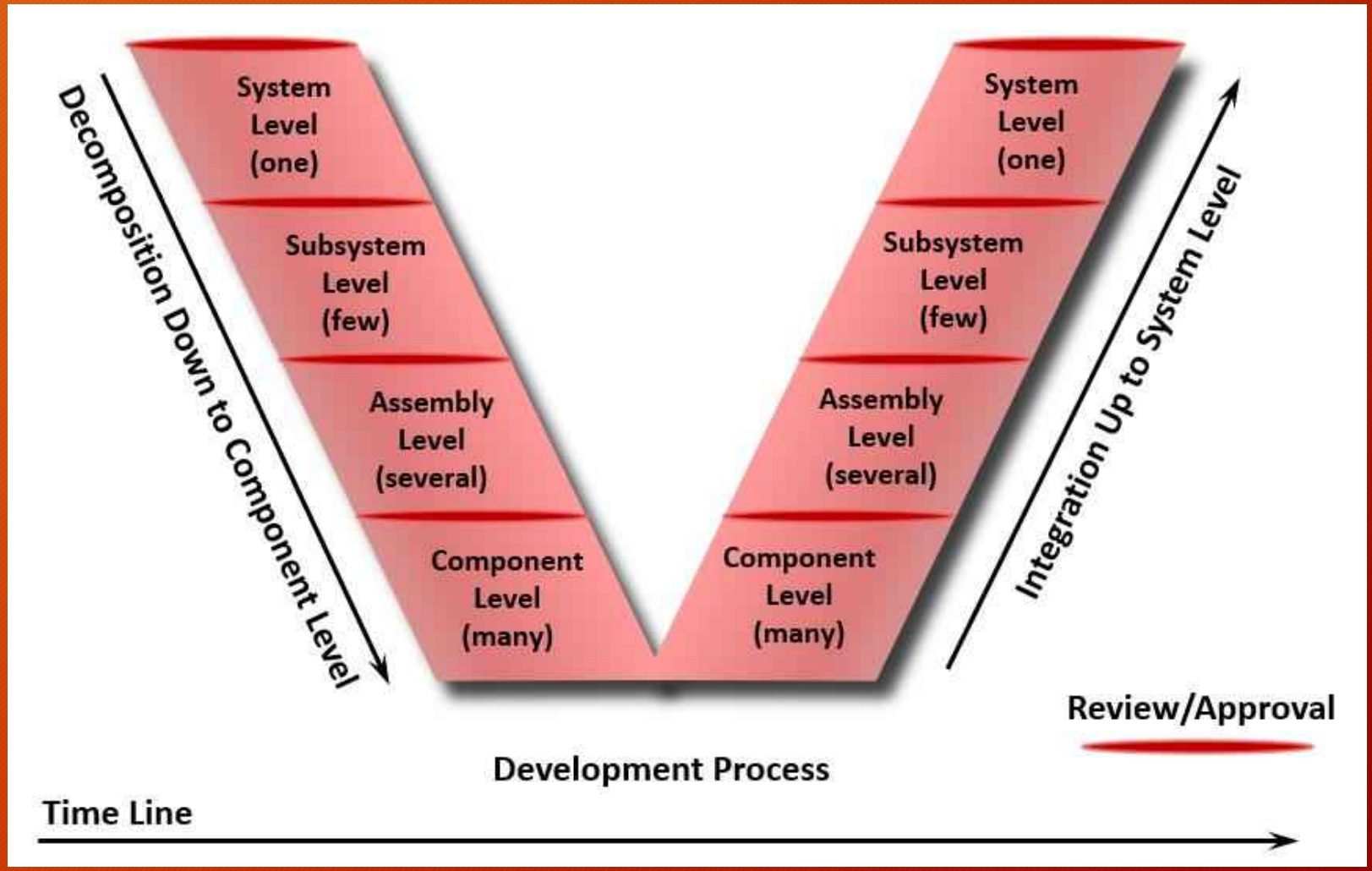
http://www.aerospaceweb.org/question/conspiracy/q0258.shtml

Decomposition of STS to Level 3 (incomplete)



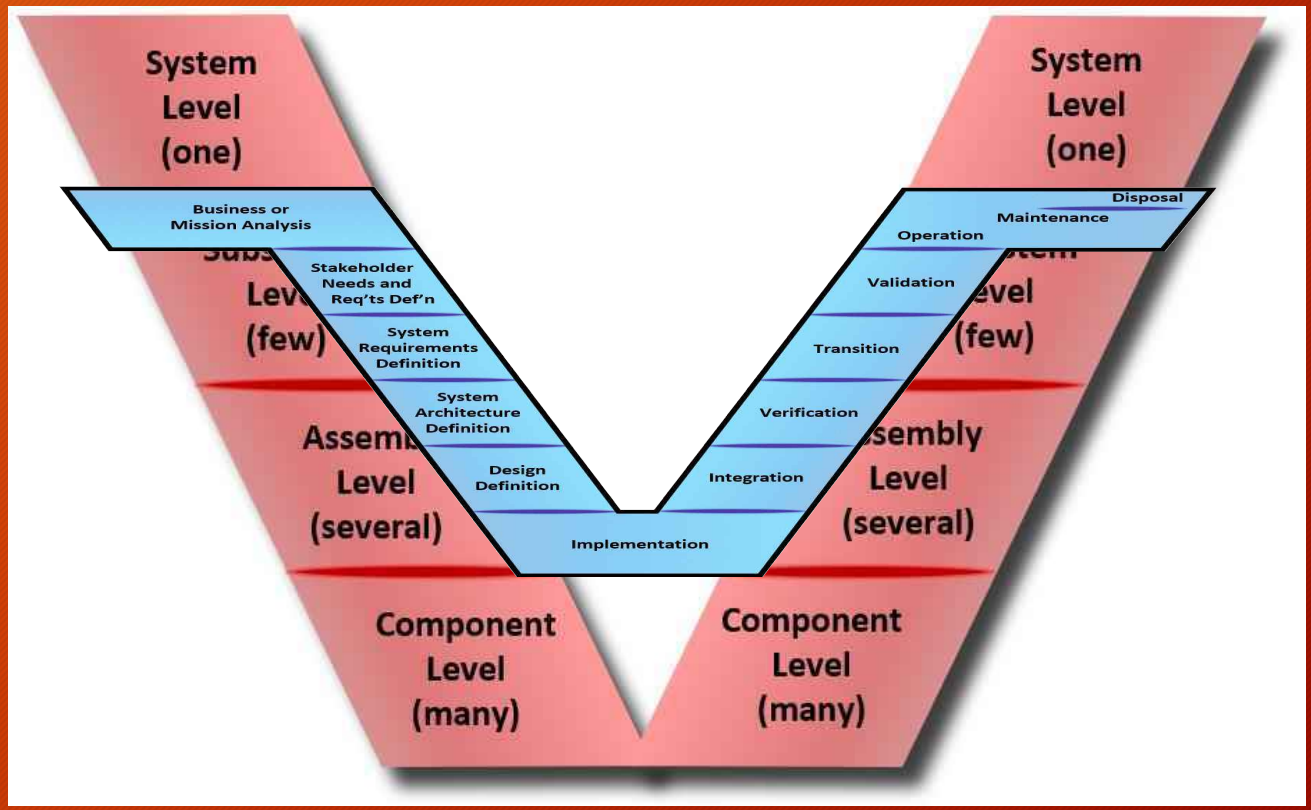
System Architecture Decomposition “VEE” Model

- Similar to how the Systems Engineering Lifecycle Processes can be laid out in a “VEE” format, so can the System Architecture Decomposition levels
- This format emphasizes the actual level of decomposition at which development work is being performed
- Instead of defining processes that are performed, the VEE emphasizes the decomposition from the System down to the Component level



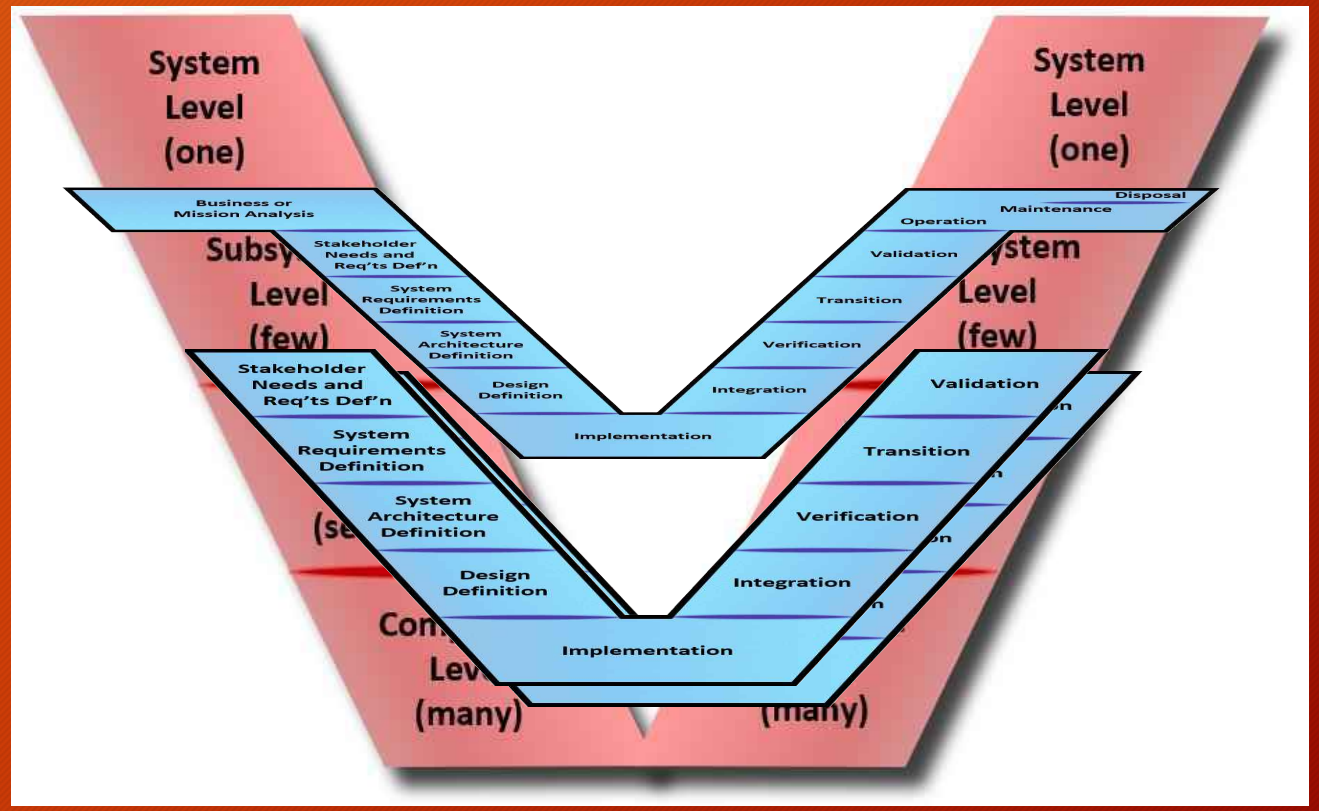
Dual “VEE” Model (System Level)

- Here, the Systems Engineering Lifecycle Processes VEE (blue) is superimposed on top of the System Architecture Decomposition VEE (red) at the System level (Level 0)
- Lifecycle processes appropriate for Level 0 (Business or Mission Analysis, etc) are indicated
- This implies that while engineers are operating at the System Level (Level 0) the SE Lifecycle processes are applied at that level



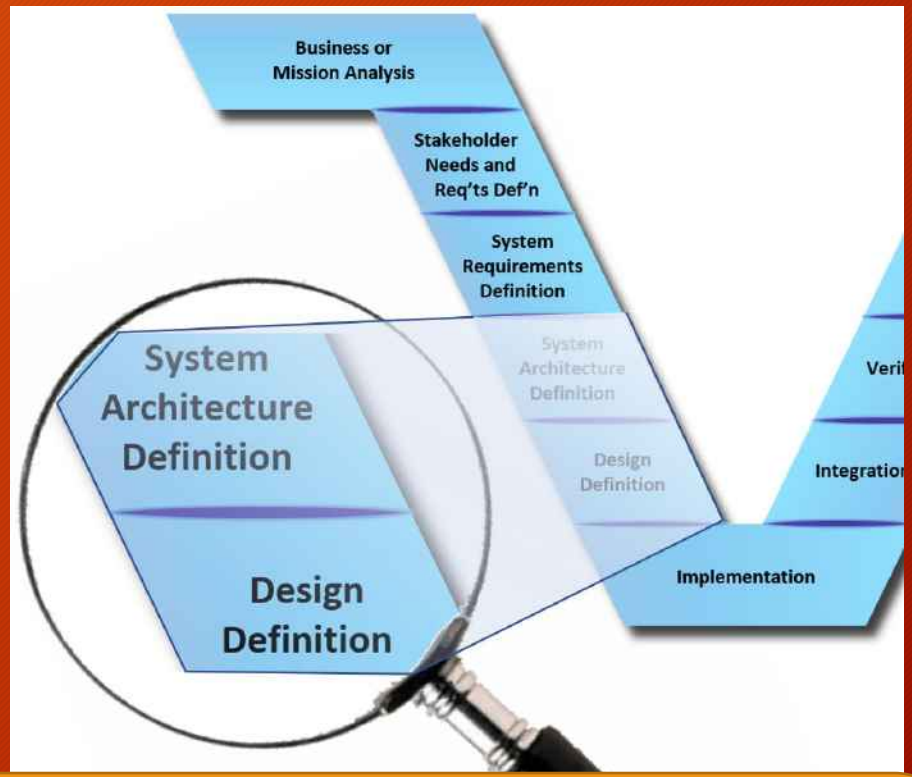
Dual “VEE” Model

- Here, the Systems Engineering Lifecycle Processes VEE (blue) is superimposed on top of the System Architecture Decomposition VEE (red) at the Subsystem level (Level 1)
- Lifecycle processes appropriate for Level 0 (Business or Mission Analysis, etc) are not included below Level 0
- This implies that while engineers are operating at the Subsystem Level (Level 1) the SE Lifecycle processes are applied at that level
- Similarly, the Systems Engineering Lifecycle Processes VEE (blue) is superimposed on top of the System Architecture Decomposition VEE (red) at all remaining lower levels of the system hierarchy



Where Does Architecting End and Design Begin?

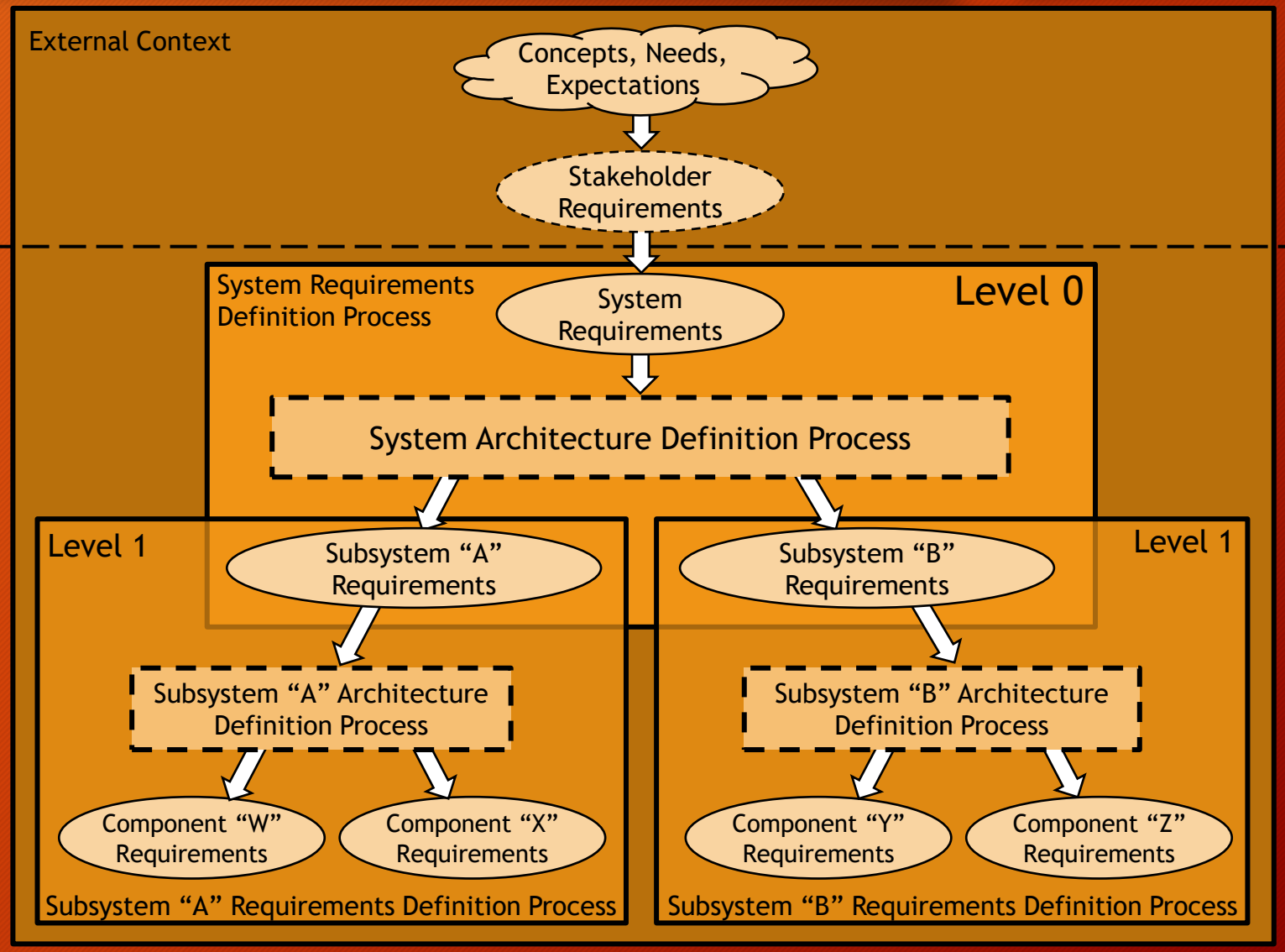
- The Systems Engineering Life Cycle Processes are iterative and recursive
- Therefore they occur repeatedly, until a review or other program-level decision is made to terminate a process
- During an iteration, however, the System Architecture Definition Process ends when:
  - A preplanned stopping point is reached
    - Due to budget, schedule, or other constraint or milestone
  - Sufficient risk and/or uncertainty has been resolved
  - Architecturally-relevant system elements can be decomposed no further
  - Other program-level decision terminates the process
- However, the complete set of artifacts needed by the Design Definition team must be made available prior to termination of the System Architecture Definition Process
- STOP architecting when the goals of the Architectural Description have been met



- An Architectural Description can be implemented in more than one way
- A detailed design can only be implemented as specified
- A design that complies with the Architectural Description is ready to proceed to implementation
- Implementation cannot begin on the basis of the Architectural Description alone

Requirements as Input Specifications to Architecture

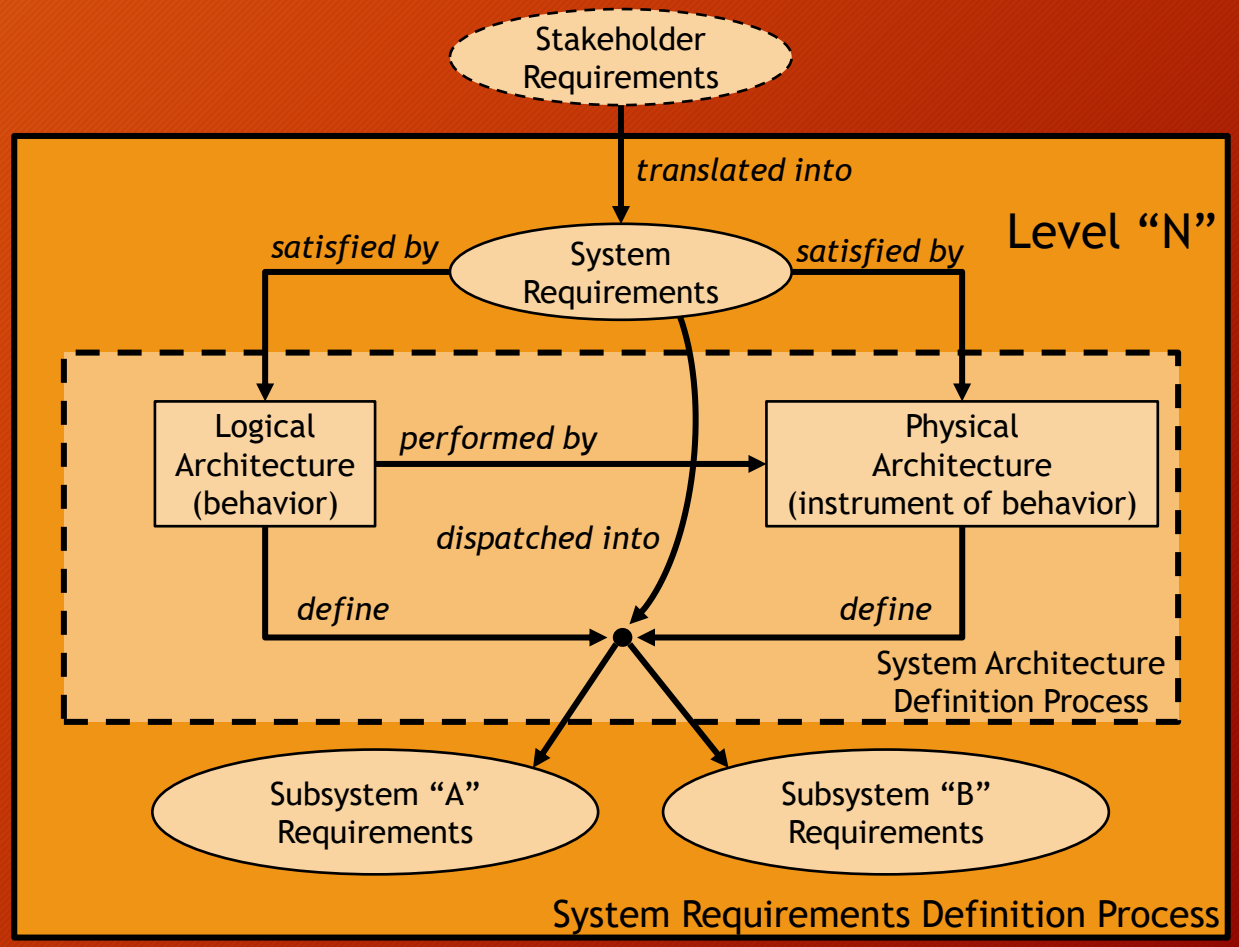
- In the Problem Definition Area
  - Concepts are developed
  - Needs are assessed
  - Expectations are voiced
  - Stakeholder Requirements are defined
- In the Solution Definition Area
  - System Requirements are defined
  - System Architecture Definition Process is implemented
  - Subsystem Requirements are defined
- The Solution Area Definition process is repeated recursively throughout the defined system structure



Adapted from Faisandier, Alan (2012)

Relation of Requirements and Architecture Processes

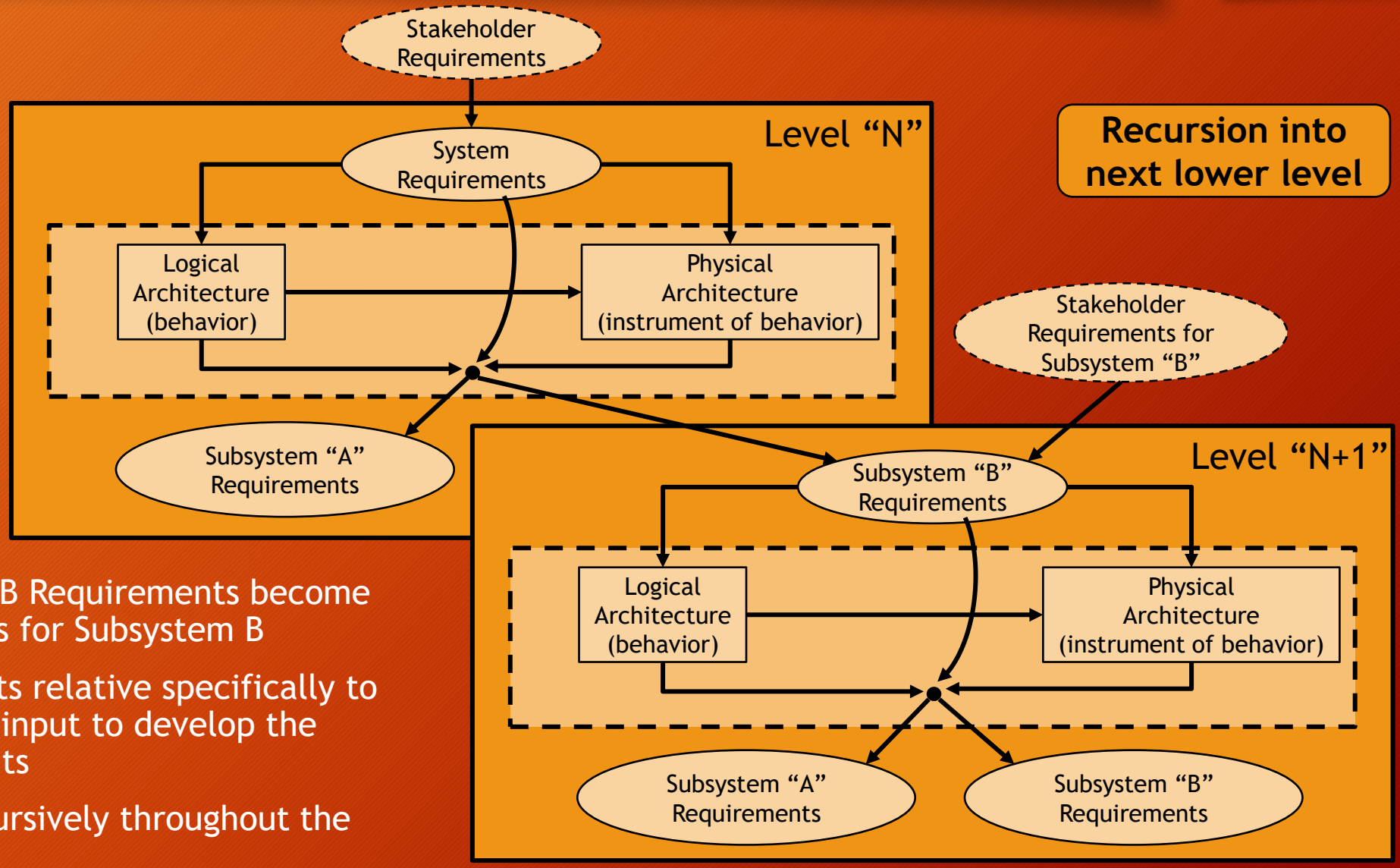
- Stakeholder Requirements are translated into System Requirements
- System Requirements are satisfied by the Logical and Physical Architectures
- The Logical Architecture (behavior) is performed by the Physical Architecture (the instrument of behavior)
- The Logical and Physical Architectures help define the requirements for the System's Subsystems (at the next lower level)
- The System Requirements are dispatched into the Subsystem Requirements
  - Read that as: the Subsystem Requirements are derived from the System Requirements
- This process is iterative in that it is performed repeatedly iteratively at the same level to refine the Logical and Physical Architectures and to refine the Subsystem Requirements
- This process is recursive in that it is performed recursively at lower levels until the architectural design objectives have been met



Adapted from Faisandier, Alan (2012)

This is a simplified view of a rather complex systems engineering management process

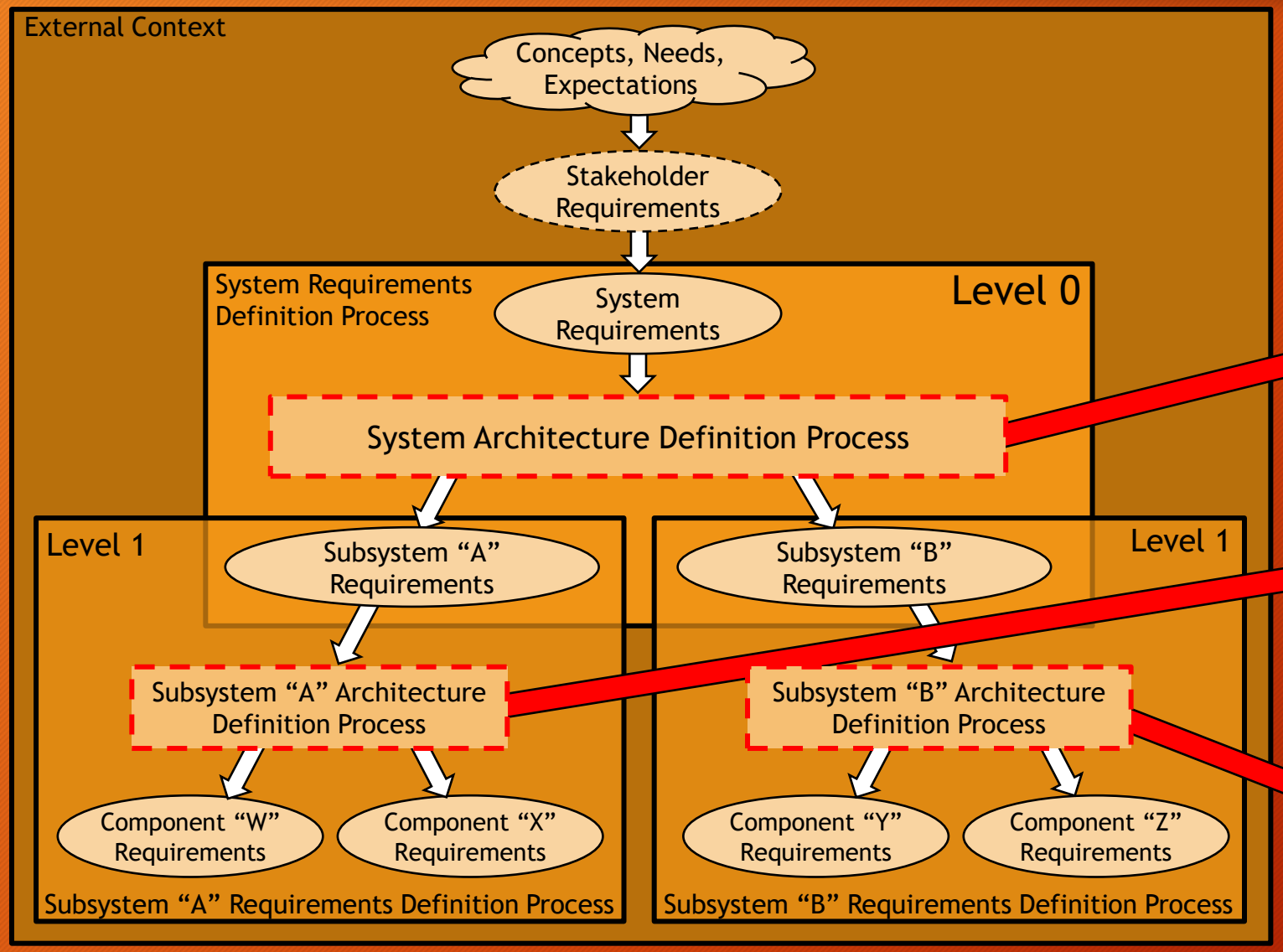
Architecture Process over Multiple Levels



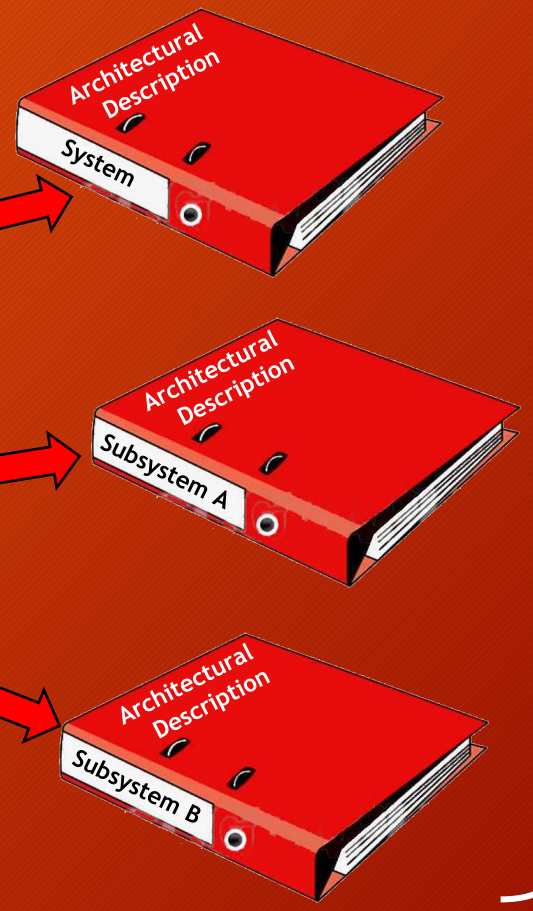
- The System's Subsystem B Requirements become the System Requirements for Subsystem B
- Stakeholder Requirements relative specifically to Subsystem B (if any) are input to develop the Subsystem B Requirements
- This process repeats recursively throughout the system structure

Adapted from Faisandier, Alan (2012)

Requirements as Input Specifications to Architecture



Adapted from Faisandier, Alan (2012)



etc...

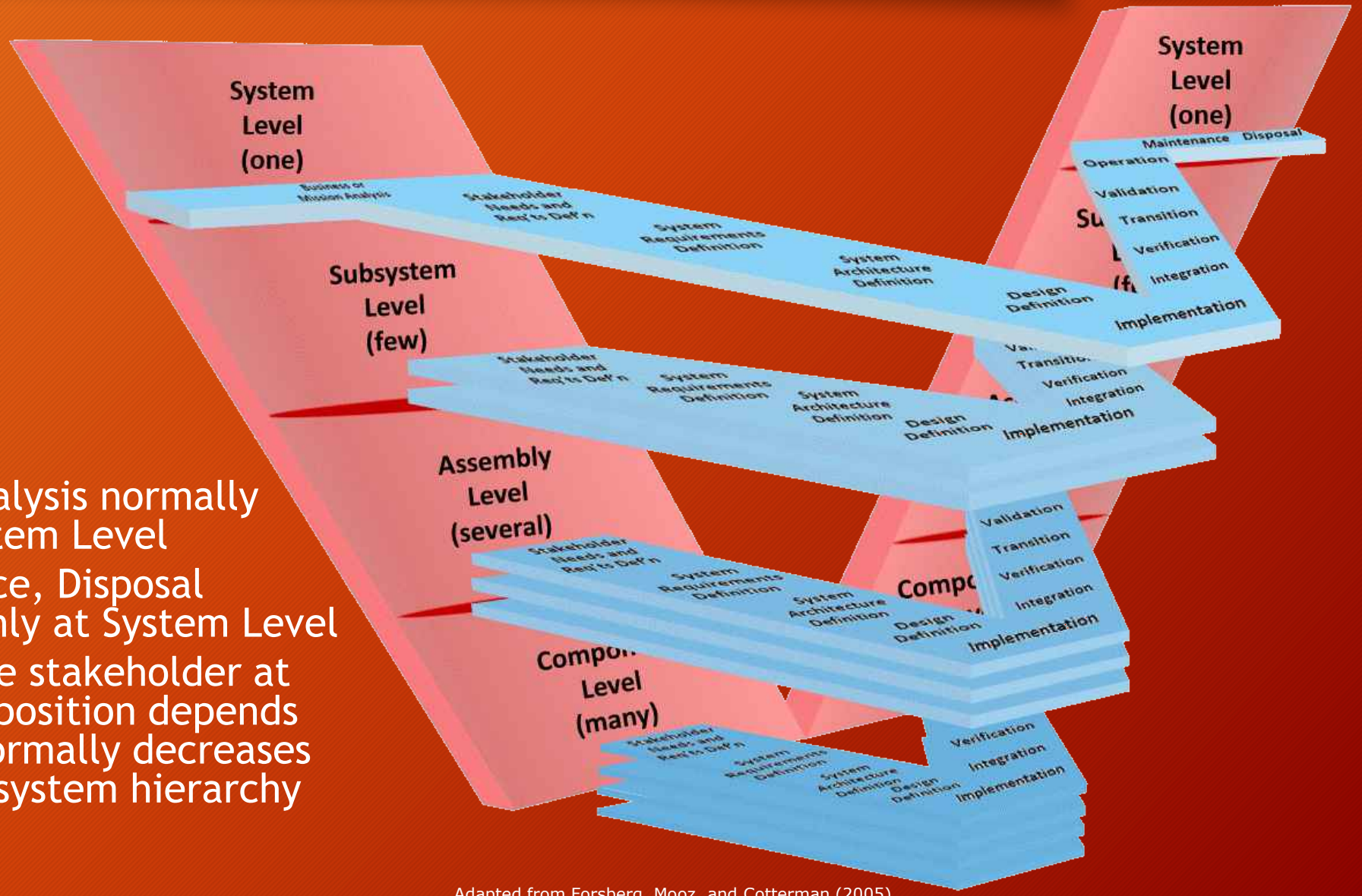
The complete set of Architecture Descriptions from all levels constitute the Total System Architecture Description

This is the principal output of the System Architecture Definition Process

And also serves as the specification (input) for the Design Definition Process

Dual-VEE

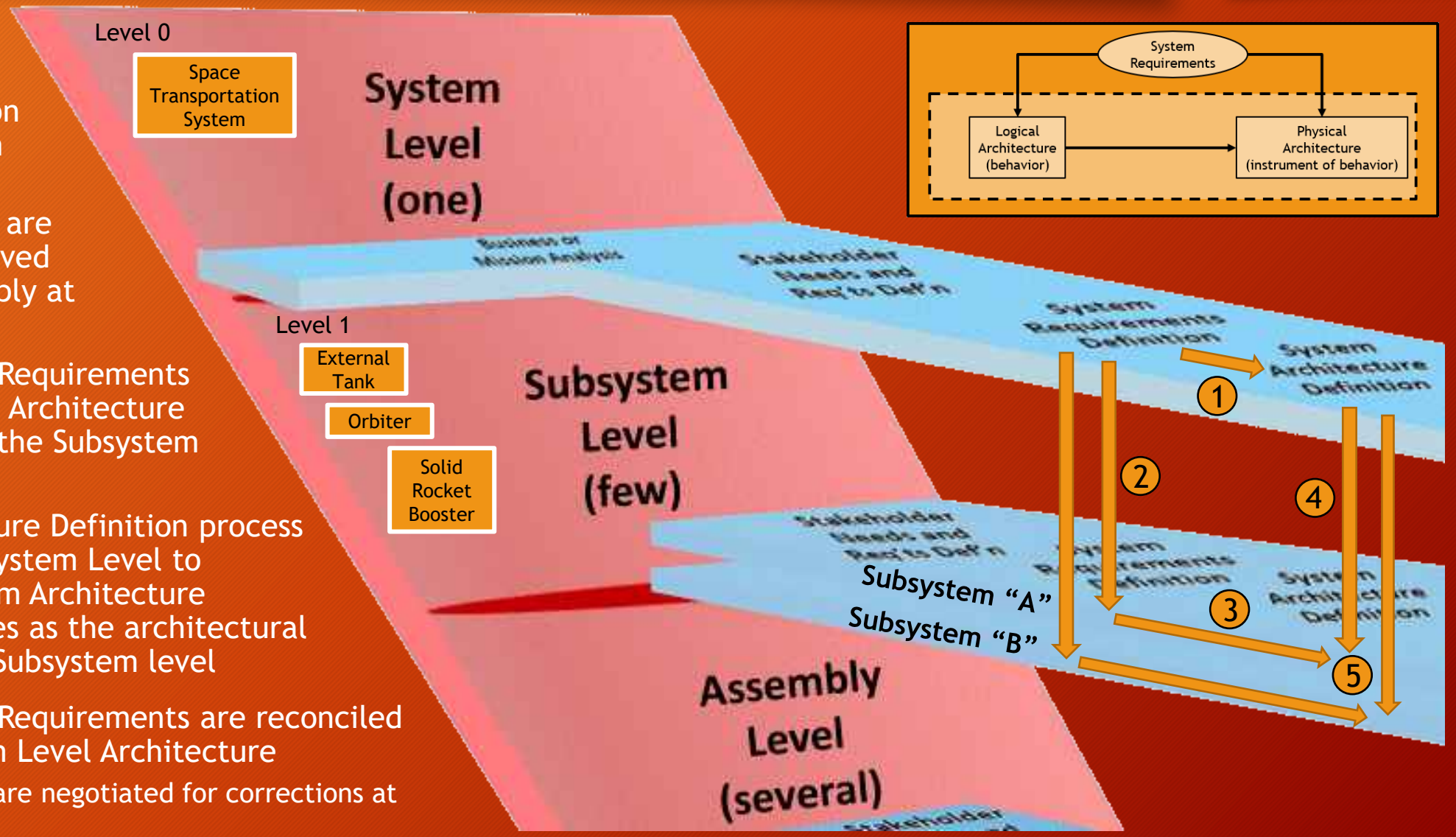
- Combines
  - Systems Engineering Lifecycle Processes VEE (blue)
  - System Architecture Decomposition VEE (red)
- Notes
  - Business or Mission Analysis normally conducted only at System Level
  - Operation, Maintenance, Disposal normally conducted only at System Level
  - The involvement of the stakeholder at lower levels of decomposition depends on the product, but normally decreases at lower levels of the system hierarchy
  - 2022



Adapted from Forsberg, Mooz, and Cotterman (2005)

Duel-VEE Interactions

1. System Requirements are fed to the System Architecture Definition process at the System Level
2. System Requirements are decomposed into derived requirements that apply at the Subsystem Level
3. The Subsystem Level Requirements are fed to the System Architecture Definition process at the Subsystem Level
4. The System Architecture Definition process is performed at the System Level to produce the Subsystem Architecture Description that serves as the architectural specifications at the Subsystem level
5. The Subsystem Level Requirements are reconciled against the Subsystem Level Architecture
   i. Any discrepancies are negotiated for corrections at the System Level



Application of Standard Processes in Industry

- Just because standard processes have been defined and have been broadly accepted across industry
  - Does not guarantee that product development will succeed
  - Some companies do not formally follow standard processes, yet survive
  - Other companies apply standard processes excessively or incorrectly and experience failures
  - The best performing companies employ and manage formal processes and do so to the correct degree appropriate to the work being performed
    - Too much or too little process is not good
    - As typical in engineering, use of processes “depends” on many factors
    - Successful companies have spent decades discovering the right balance of application of process to the development jobs they perform
- At a minimum, engineering management and engineering staff must be familiar with standard SE processes
- Project team management should decide the right balance of process to employ on the project
- The project should publish a Systems Engineering Management Plan (SEMP) to define exactly which processes are being applied to what degree

Architecture: Art and Science

- Every human-built system has an architectural description
  - On paper
  - In a model, or a repository
  - In someone's head
  - Lost to history
  - Possibly recoverable by performing reverse engineering of the system
- The process of establishing the architecture of a system is both an art and a science
  - The artistic aspect is addressed by employing proven heuristics (rules of thumb)
  - The scientific aspect is addressed by following documented principles, procedures, processes, methods, using the correct tools, etc

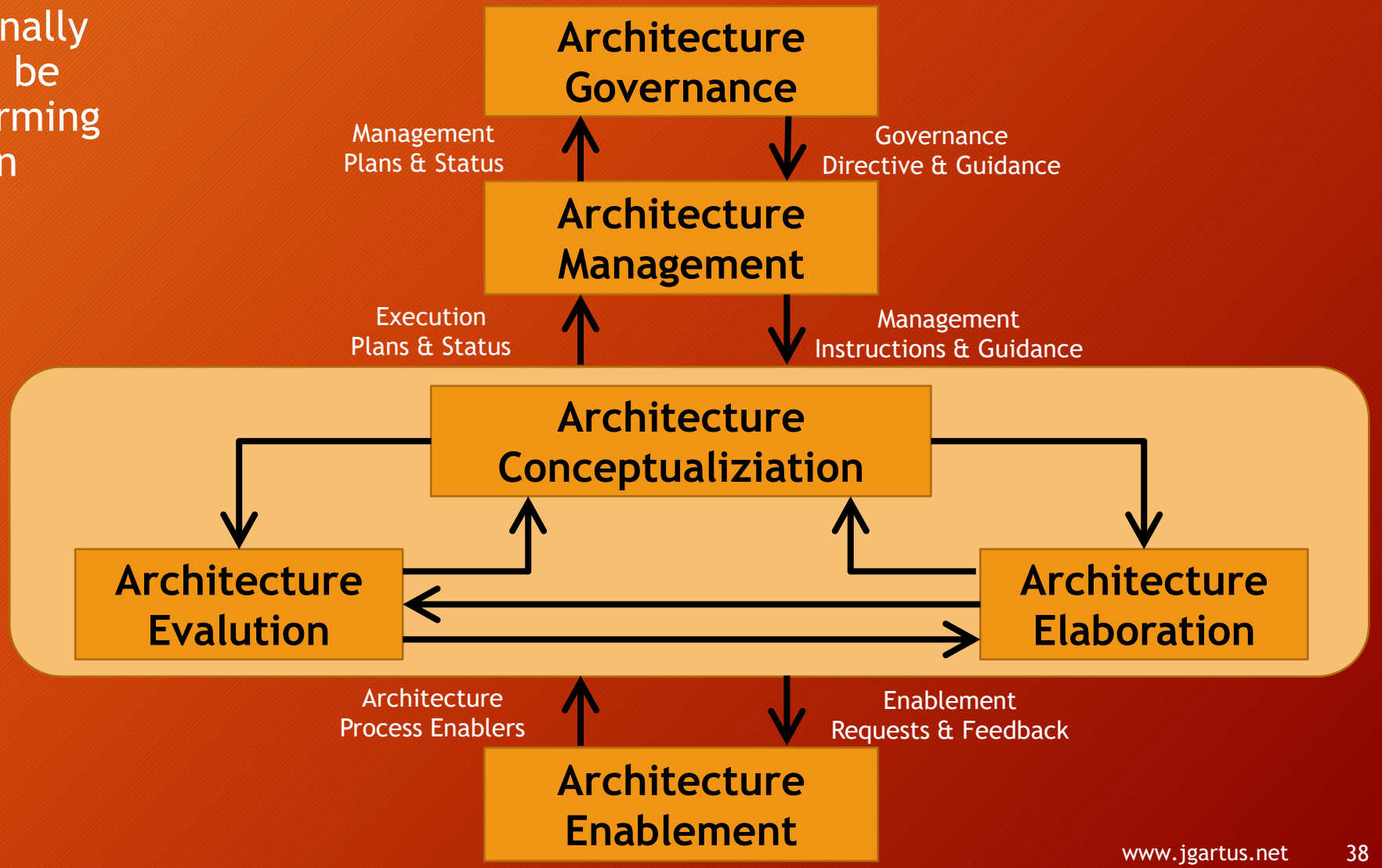
Principles, Methods, Tools

- Principles are the underlying and long-enduring fundamentals
  - Nearly always valid
  - Includes a prescriptive part
  - Includes a descriptive part
- Methods are the ways of organizing approaches and tasks to achieve a concrete end
  - Should be solidly grounded on principles
  - Are usually applicable to the task being performed
- Tools are the contemporary ways to facilitate process
  - They enable the employment of methods and the compliance with principles
  - They can also sometimes constrain the methods used and the principles that can be followed

Source:
- System Architecture; Edward Crawley, Bruce Cameron, Daniel Selva; 2016; Pearson Higher Education, Inc

Architecture Definition Processes

- ISO/IEC/IEEE 42020 Software, systems and enterprise — Architecture processes
  - Defines the internationally accepted processes to be employed when performing Architecture Definition



Architecture Governance Subprocess

Architecture Governance

1. Prepare for and plan the architecture governance effort
2. Monitor, assess, and control the architecture governance activities
3. Establish architecture collection objectives
4. **Make architecture governance decisions**
5. **Monitor and assess compliance with governance directives and guidance**
6. **Review implementation of governance directives and guidance**

Architecture Management Subprocess

Architecture Management

1. Prepare for and plan the architecture management effort
2. Monitor, assess and control the architecture management activities
3. Develop architecture management approach
4. **Perform management of the architecture collection**
5. **Monitor architecting effectiveness**
6. **Prepare for completion of the architecture management plan**

Architecture Conceptualization Subprocess

Architecture Conceptualization

1. Prepare for and plan the architecture conceptualization effort
2. Monitor, assess and control the architecture conceptualization activities
3. **Characterize problem space**
4. **Establish architecture objectives and critical success criteria**
5. **Synthesize potential solution(s) in the solution space**
6. **Characterize solutions and the tradespace**
7. **Formulate candidate architecture(s)**
8. **Capture architecture concepts and properties**
9. Relate the architecture to other architectures and to relevant affected entities
10. Coordinate use of conceptualized architecture by intended users

Architecture Evaluation Subprocess

Architecture Evaluation

1. Prepare for and plan the architecture evaluation effort
2. Monitor, assess and control the architecture evaluation activities
3. Determine evaluation objectives and criteria
4. Determine evaluation methods and integrate with evaluation objectives and criteria
5. Establish measurement techniques, methods and tools
6. Collect and review evaluation-related information
7. Analyze architecture concepts and properties and assess stakeholder value
8. Characterize architecture(s) based on assessment results
9. Formulate findings and recommendations
10. Capture and communicate evaluation results

Architecture Elaboration Subprocess

Architecture Elaboration

1. Prepare for and plan the architecture elaboration effort
2. Monitor. assess and control the architecture elaboration activities
3. **Identify or develop architecture viewpoints**
4. **Develop models and views of the architecture(s)**
5. **Relate the architecture to other architectures and to relevant affected entities**
6. Assess the architecture elaboration
7. Coordinate use of elaborated architecture by intended users

Architecture Enablement Subprocess

Architecture Enablement

1. Prepare for and plan the architecture enablement effort
2. Monitor, assess, and control the architecture enablement activities
3. Manage the architecture process enablers
4. Acquire, develop and establish enabling capabilities, services and resources
5. Deploy enabling capabilities, services and resources
6. Improve architecture enablement capabilities, services and resources

Lessons Learned

- Over time, we have learned
  - Bad architectural decisions can kill a large project from the outset
  - Systems embodying the mistakes of the past do not survive
  - Those that embody sound architectures generally do survive — and even prosper
    - However, there is no guarantee
    - Often, getting the architecture “right” merely creates a platform on which the execution of the product can then either flourish or flounder
- A good system architecting process allows us to
  - Look downstream and identify which constraints and opportunities will be central to delivering value to the stakeholder
  - Perform trades among several architectural concepts early while the cost of doing so is low
  - Focus on “emergence” where entity functionality comes together to produce new emergent functionality where it did not exist before

Source:
- System Architecture; Edward Crawley, Bruce Cameron, Daniel Selva; 2016; Pearson Higher Education, Inc

Early Decision Making

- It is important to make key decisions early on in the architectural process
  - Unfortunately, many of these early architectural decisions must be made without full knowledge of the system's eventual scope
    - Some of these may be the infamous "known unknowns"
  - Yet, such decisions need to be made in order for other design work to proceed
  - These early decisions will have enormous impact on the eventual design
- The good news is that these early decisions can be analyzed and treated
  - The architecture of the system merits careful analysis and scrutiny
    - Despite uncertainty around scope
    - And even without knowing the detailed design of components
- Focus on decision making
  - System architectures are rarely chosen in one fell swoop
    - They are iteratively defined by a series of choices
  - A formal decision-making process enables system architects to directly trade the choices for each decision, rather than the underlying designs they represent
  - This encourages broader concept evaluation
  - And to enables system architects to order decisions according to their leverage on the system performance

Source:
- System Architecture; Edward Crawley, Bruce Cameron, Daniel Selva; 2016; Pearson Higher Education, Inc

Skills Required of a Good System Architect

- Use systems thinking in both a product context and a system context
- Analyze and critique the architecture of existing systems
- Identify architectural decisions, and differentiate between architectural and non-architectural decisions
- Create the architecture of new or improved systems, and produce the deliverables of the architecture
- Place the architecture in the context of value and competitive advantage for the product and the firm
- Drive out the ambiguity inherent in the upstream process by
  - Defining the context and boundaries of the system
  - Interpreting needs
  - Setting goals
  - Defining the externally delivered functions

Source:
- System Architecture; Edward Crawley, Bruce Cameron, Daniel Selva; 2016; Pearson Higher Education, Inc

Skills Required of a Good System Architect

- Create the concept for the system, consisting of internal function and form, while thinking holistically and out of the box when necessary
- Manage the evolution of system complexity and provide for future uncertainty so that goals are met and functions are delivered
- Ensure the system remains comprehensible to all during its design, implementation, operation, and evolution
- Challenge and critically evaluate current modes of architecting
- Identify the value of architecting, analyze the existing product development process of a firm, and locate the role of architecting in the product development process
- Develop the guiding principles for successful architecting

Source:
- System Architecture; Edward Crawley, Bruce Cameron, Daniel Selva; 2016; Pearson Higher Education, Inc

References

1. Crawley, Edward; Cameron, Bruce; Selva, Daniel (2016). *System Architecture: Strategy and Product Development for Complex Systems*, Pearson Higher Education Inc, Hoboken, NJ
2. Dang, Ahn (2019). Known Knowns, Unknown Knowns, and Unknown Unknowns. Retrieved from https://medium.com/the-world-in-the-future/known-knowns-unknown-knowns-and-unknown-unknowns-b35013fb350d
3. ISO/IEC/IEEE 15288 Systems and software engineering — System life cycle processes (2015). https://www.iso.org/standard/63711.html
4. SEBoK Editorial Board. 2021. The Guide to the Systems Engineering Body of Knowledge (SEBoK), v. 2.4, R.J. Cloutier (Editor in Chief). Hoboken, NJ: The Trustees of the Stevens Institute of Technology. Accessed October 2021. www.sebokwiki.org
5. Weinmeyer, Warren (2012). *An Introduction to Fundamental Architecture Concepts*. Retrieved from https://www.academia.edu/8299805/An_Introduction_to_Fundamental_Architecture_Concepts
6. Martin, James N. (2017). *Overview of an Emerging Standard on Architecture Evaluation – ISO/IEC 42030.* Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2334-5837.2017.00418.x
7. Martin, James N. (2018). *Overview of Emerging ISO Standards on Architecture.* Retrieved from http://www.put_a_working_URL_here.com
8. Faisandier, Alan (2012). *Systems Architecture and Design*, Sinergy'Com, Belberaud, France
9. Forsberg, Kevin; Mooz, Hal; Cotterman, Howard (2005). *Visualizing Project Management: Models and Frameworks for Mastering Complex Systems*, 3rd Edition, John Wiley & Sons, Inc., Hoboken, New Jersey