Systems Engineering for DoD

Lesson 6

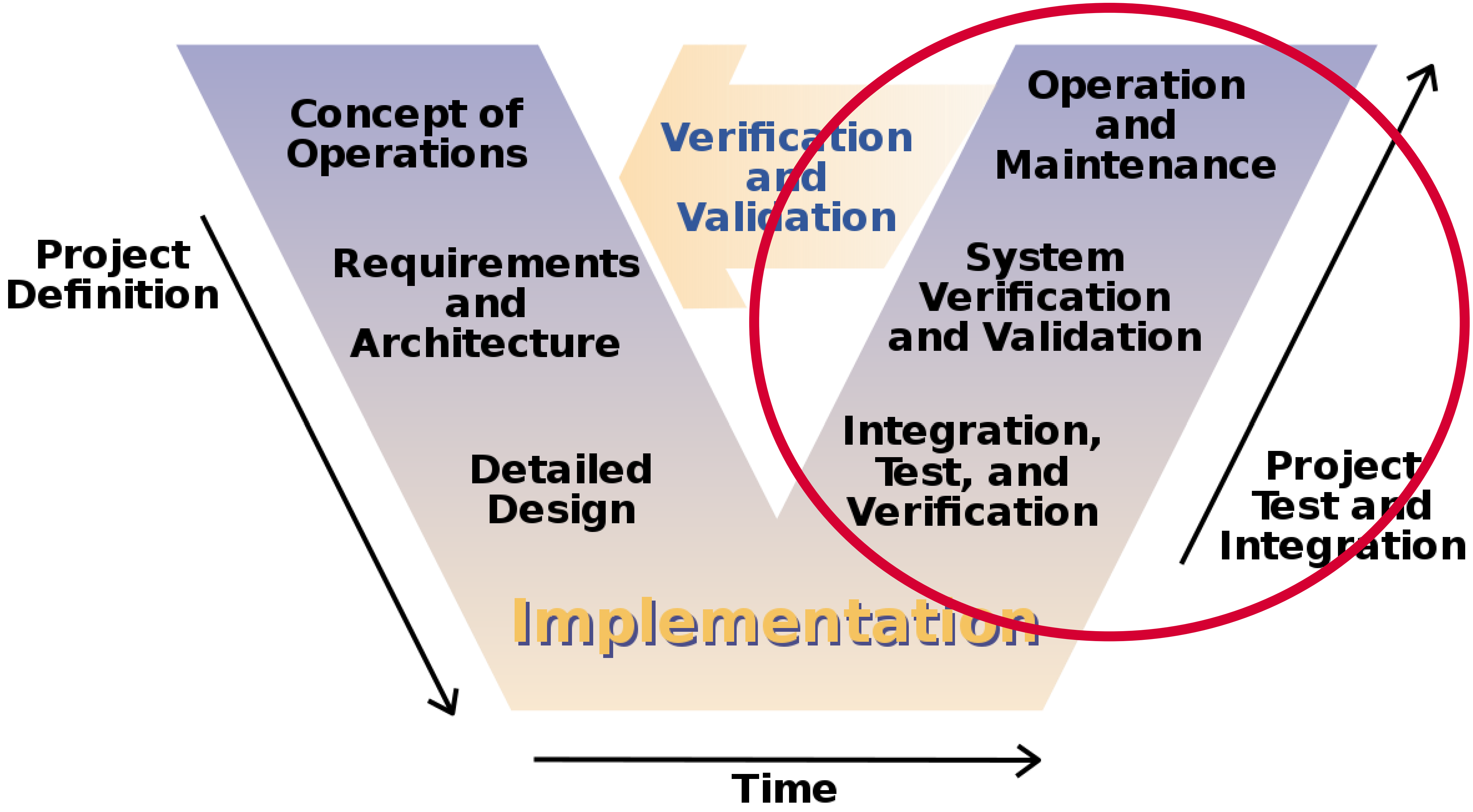
Integration

What does "System Integration" mean?

Image by Gerd Altmann from Pixabay

How INCOSE defines Integration:

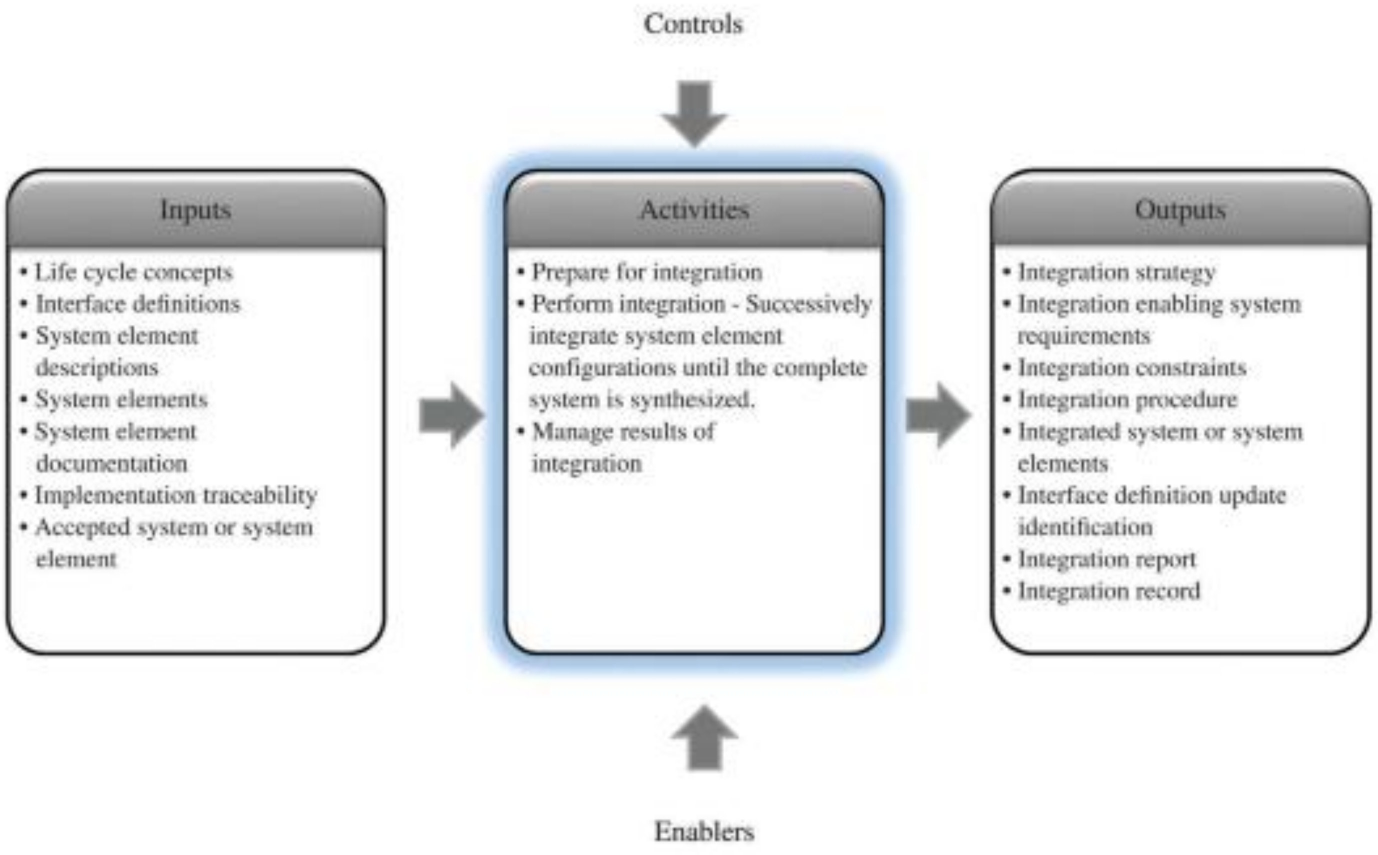
To synthesize a set of system elements into a realized system, product or service that satisfies system requirements, architecture and design.


Concept of Operations
Requirements and Architecture
Detailed Design
Implementation
Integration, Test, and Verification
System Verification and Validation
Operation and Maintenance
Verification and Validation
Project Definition
Project Test and Integration
Time

Integration process

IEEE 15288, [6.4.8.1] The purpose of the Integration process is to synthesize a set of system elements into a realized system (product or service) that satisfies system requirements, architecture, and design.

Integration process



NASA INTEGRATION PROCESS

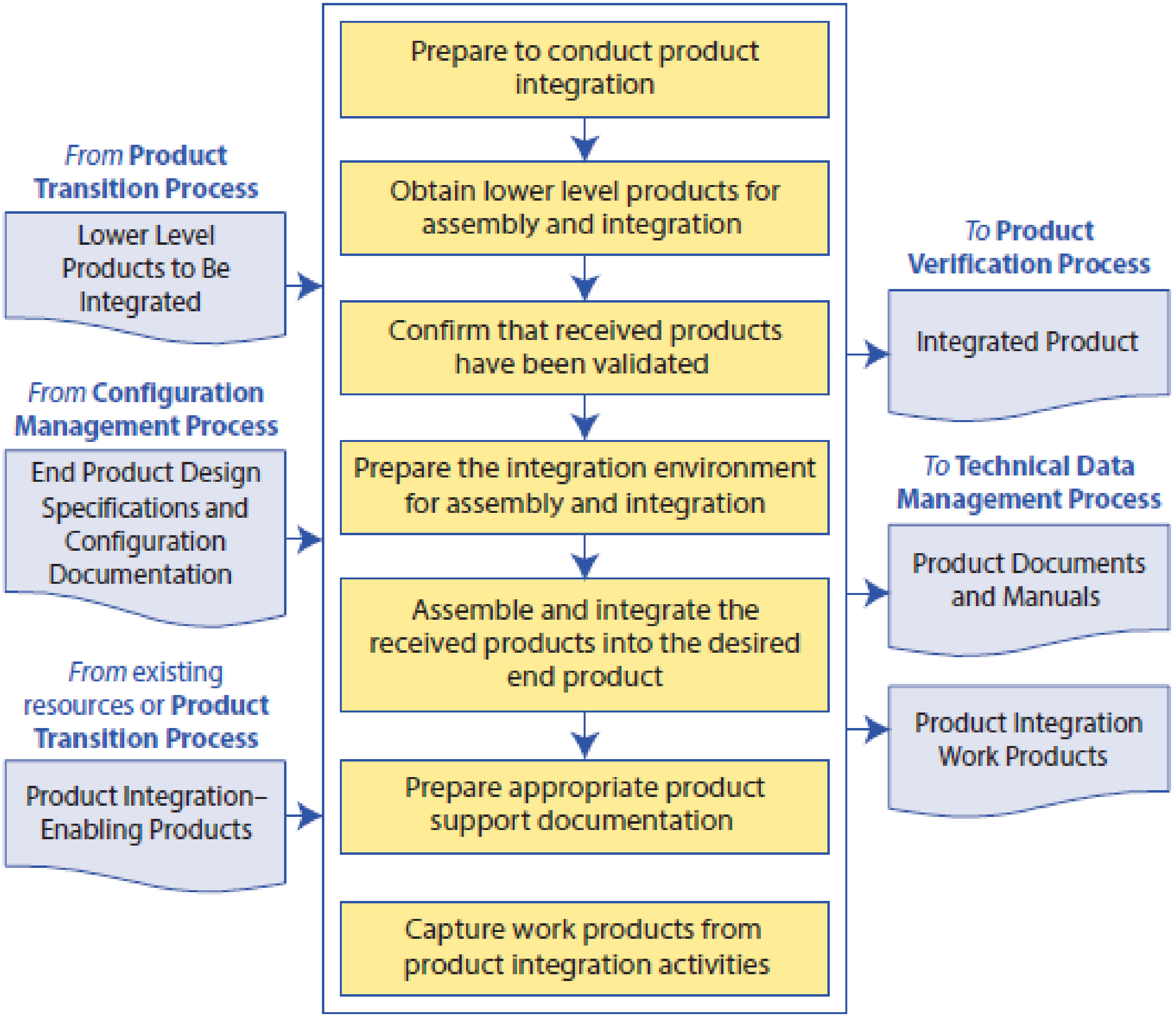


FIGURE 5.2-1 Product Integration Process

-NASA Systems Engineering Handbook

Systems Integration

Logical

- Develop information systems that allow organizations to share data with all of its stakeholders based on need and authorization.
- Management needs to change organizational structures, processes, and employee roles and responsibilities.

Physical

- Provide seamless connectivity between heterogeneous systems.
- Operational process reengineering involves changing the mindset of the employees in the organization, encouraging and enabling them to do their tasks in a new way.

Analyses that you should report out on:

- Implementation plan (order, schedule) with rationale, based on the selected implementation alternatives
- How the team is confident that the proposed design will satisfy requirements and align with architecture across all system lifecycle phases
- What issues, if any, with previous work performed were discovered in this phase and how those issues were addressed
- What risks to downstream activities were identified and how they are being or will be addressed

Artifacts that you ***may*** choose to use to show this:

- Physical system element identification & rationale (e.g., N2 Chart)
- Interface diagram defining physical system elements and interfaces
- Physical system element design characteristics and/or constraints
- Physical system element interface definitions
- Build/Buy/(Re)use plan for identified physical system elements with rationale (e.g., Trade Study)
- Traceability between system elements and IFs and requirements/logical architecture and/or guidelines/principles

Interface Testing

Needed whenever modules or subsystems are combined to create a larger system

Goal is to identify faults due to interface errors or to invalid interface assumptions

Particularly important in object-oriented systems development

Interface Types

Parameter interfaces

- data passed normally between components

Shared memory interfaces

- block of memory shared between components

Procedural interfaces

- set of procedures encapsulated in a package or sub-system

Message passing interfaces

- sub-systems request services from each other

Interface Errors

Interface misuse

- parameter order, number, or types incorrect

Interface misunderstanding

- call component makes incorrect assumptions about component being called

Timing errors

- race conditions and data synchronization errors

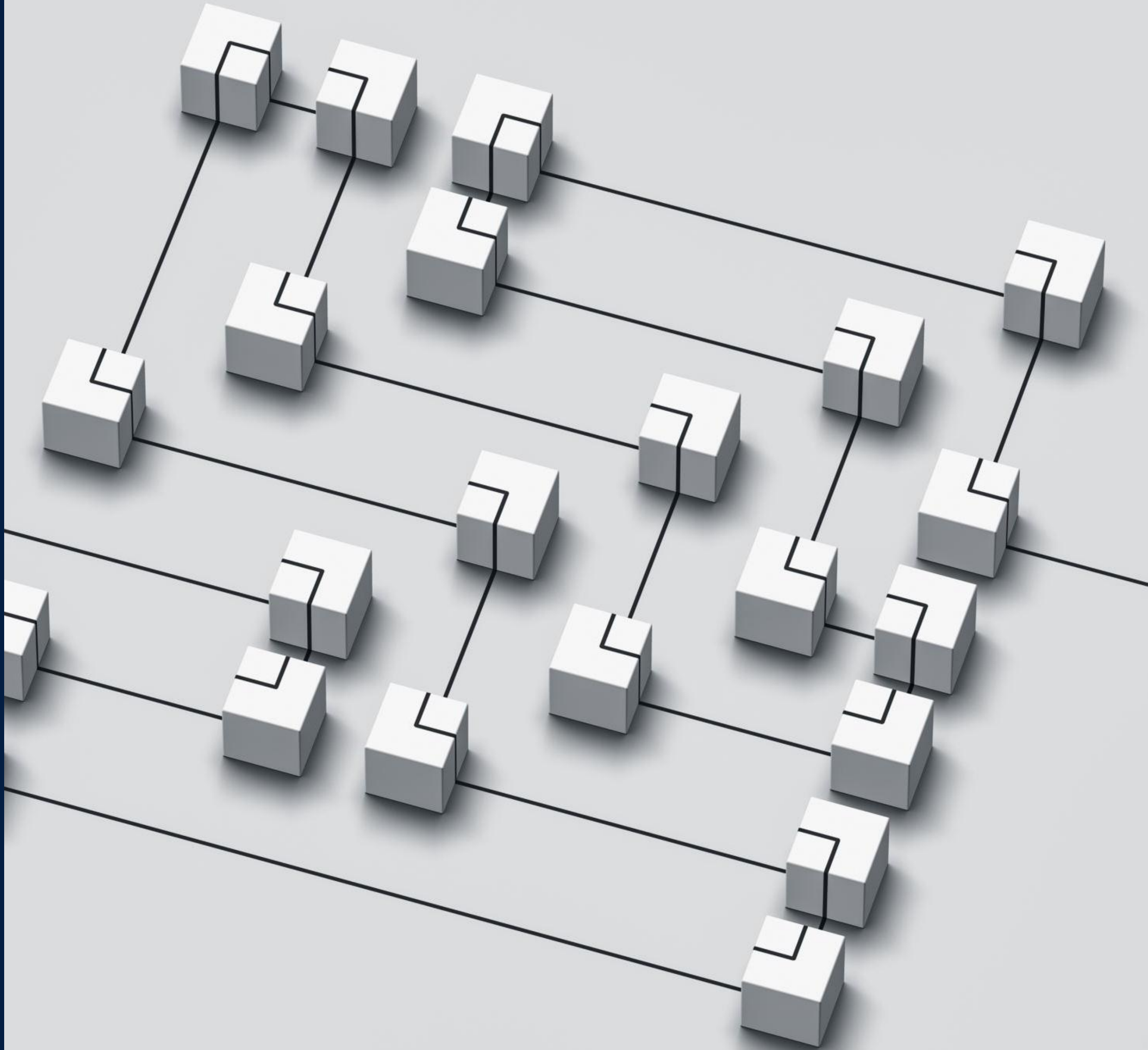
Systems Integration

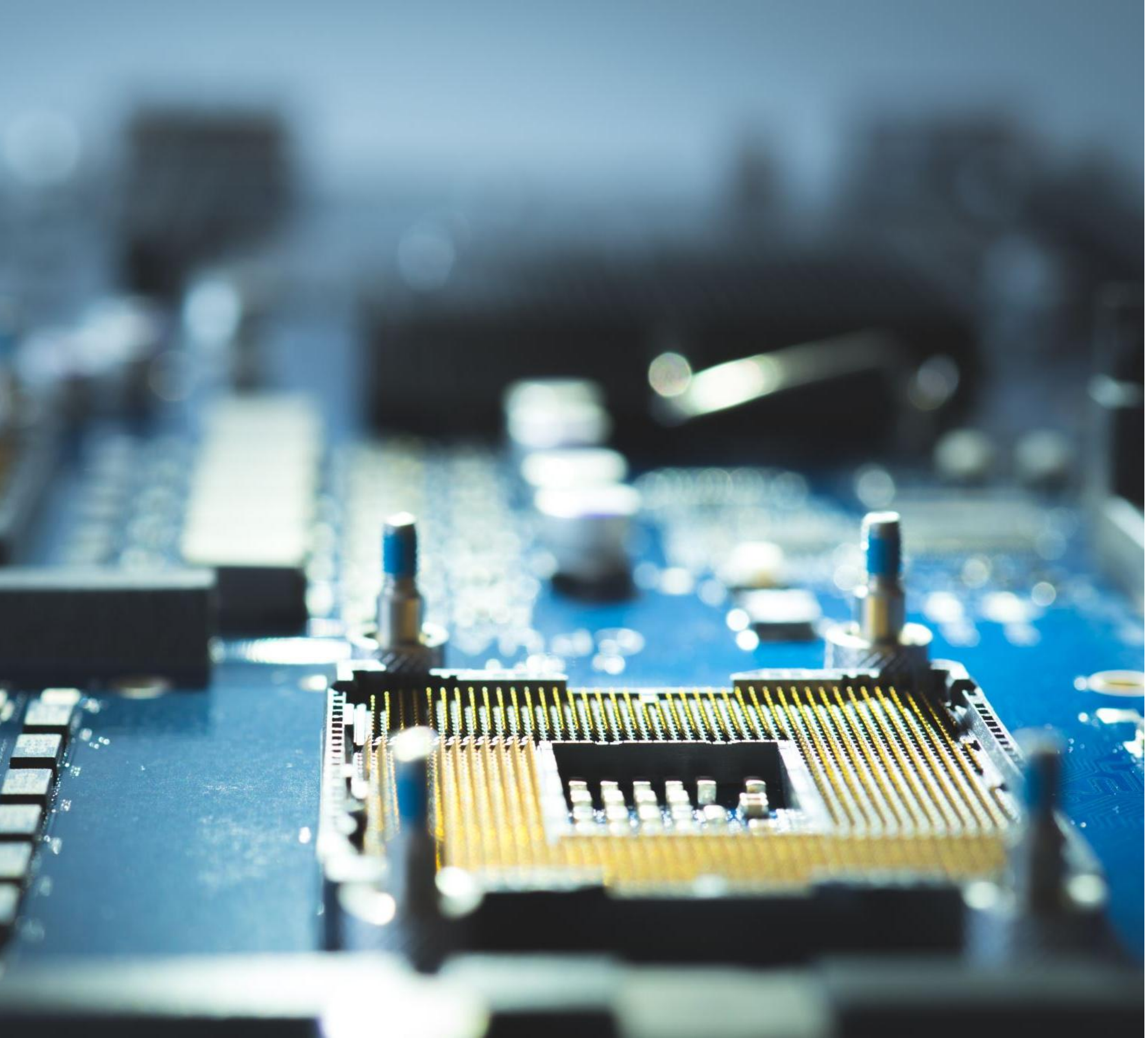
Logical

- Develop information systems that allow organizations to share data with all of its stakeholders based on need and authorization.
- Management needs to change organizational structures, processes, and employee roles and responsibilities.

Physical

- Provide seamless connectivity between heterogeneous systems.
- Operational process reengineering involves changing the mindset of the employees in the organization, encouraging and enabling them to do their tasks in a new way.





- Physical architecture
- (represented in diagrams, tables, descriptions)

  - System diagrams / System interface descriptions
  - Network topologies
  - System UI / UX prototypes
  - State charts
  - Rule models
  - Interface control documents/diagrams (ICD)

- Traceability between the physical architecture and requirements and logical architectures
- Issues with work done to date and plans to address them, e.g.:
    - If the design can't meet requirement(s)…
    - If the implementation will far exceed a requirement…
    - If technology has advanced such that it impacts the original solution concept.

Acquisition

What risks are associated with the implementation plan and how will you address them?

Has the team considered, based on the implementation plan and physical system elements/components, risks to integration, verification, validation…? (E.g., storage, facilities, equipment, personnel, environments, etc.)

Given the implementation plans, is the project plan (including downstream activities: integration, V&V, deployment, etc.) and schedule still feasible or does it need to be adjusted?

The integration plan defines:

DEFINES THE ORDER OF ASSEMBLY

DEFINES THE AGGREGATE COMPOSITION

DEFINES AGGREGATES TO BE VERIFIED

DEFINES THE ASSEMBLY PROCESS (AS NECESSARY)

IDENTIFIES ENABLERS NEEDED AND THE PLAN TO ACQUIRE THEM

- Where is assembly taking place?
- Who is involved in the assembly?
- How long will assembly take?
- When is the assembly going to occur (possibly ‘time after delivery from implementation’)?
- Are there special equipment, facilities, resources needed for assembly; what are they?
- Are there dependencies on other unavailable components; what are they?
- Are there dependencies on unavailable external systems, elements; What are they?

Global Integration
All delivered elements are assembled in one step.

☺ Does not require simulation or emulation of unavailable system elements.

☹ Difficult to isolate faults.

Best for simple systems with few internal interactions, few system elements, and low technical risk.

"With the stream" Integration

System elements are integrated as they become available.

Enables a quick start to integration.

Can be complex because unavailable elements are simulated or emulated. Global tests left late in schedule.

Best for well-controlled systems with low technical risk.

Incremental Integration

One (or a few) elements are added to an already integrated set of elements.

Allows for fast fault localization.

Requires simulators or emulators for unavailable or unintegrated elements.

Can be used with any system architecture

Subset Integration

Elements are assembled into aggregates, verified aggregates are assembled.

☺ Enables parallel integration to save time. Enables partial system delivery.

☹ Requires clearly defined functional chains.

Best for architectures with clearly delineated functional chains

Top-down Integration

Elements are integrated in "activation" order.

Enables fast available of a system "skeleton."
Enables early detection of architectural faults.

Requires creation of stubs until lower level functions are integrated.

Best for software systems where underlying "leaf-level" functions can be stubbed…

Bottom-up Integration

Elements are integrated in reverse "activation" order.

☺ Enables early detection of faults.
Number of simulators and/or stubs is reduced.

☹ Late detection of architectural faults.

Good for software or hardware-intensive systems

Criterion-based Integration

Most critical elements (e.g., risky, innovative, priority) are integrated first.

☺ Enables early testing of critical / risk-prone elements.
Early verification of arch. & design choices

☹ Can be difficult to define aggregates and effectively test system as it will be used.

Good for systems with high risk elements

Tips for selecting an integration strategy:

Account for the scheduled availability of system elements

Account for the planned deployment of system elements

Account for the availability of integration enablers

Integrate in an order the allows for fault detection more easily

Account for system element risk