Defining System Development Lifecycles to Plan and Manage Projects Effectively

Bruce A. Boyd

Associate Technical Fellow

The Boeing Company

Problem Statement

Many plans for system development projects do not reflect an understanding of the development life-cycle strategies for the system elements being defined, developed and integrated.

This frequently leads to:

- Simple, success-oriented (i.e., fantasy) plans
- Frequent project rebaselining and replanning
- Delivering less than desired capability
- Adding additional development cycles
- Work-arounds for late subsystems and components
- Re-architecting the system

Proposed Solution

- Identify development life-cycle strategies for the overall system and for each major subsystem and component being developed.

- Integrate these life-cycle strategies into the plans for developing each subsystem and component comprising the system.

A Life-Cycle-Based Planning Process

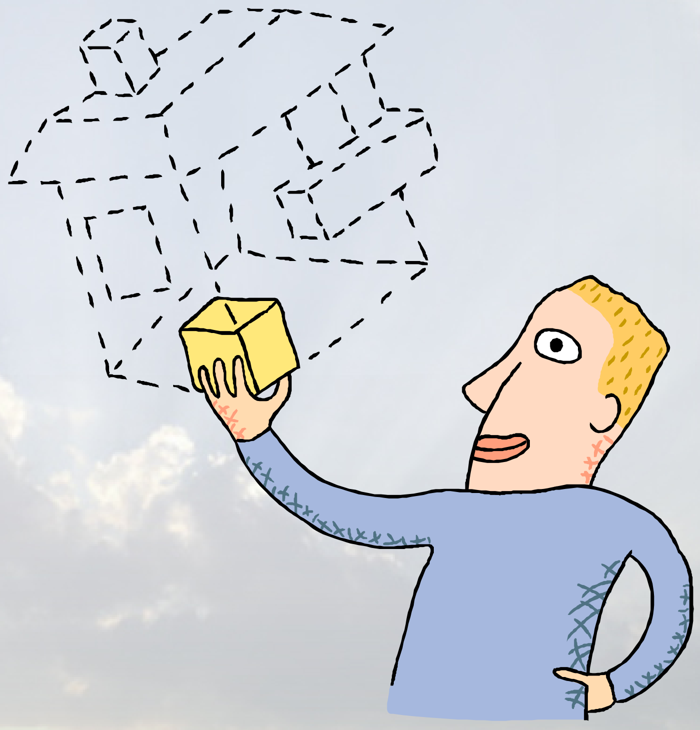
1. Define the overall system architecture
2. Identify the applicable life-cycle strategies
3. Define the appropriate life-cycle phases
4. Define the iterations for each development
5. Identify the allocation and integration events
6. Identify the processes and work products
7. Define the overall project plan and schedule (e.g., IMP and IMS)
8. Define tasks, estimates, staffing, etc.

1. Define the System Architecture

Based upon an initial review and analysis of the customer's desired capabilities, define an overall system architecture (design concept).

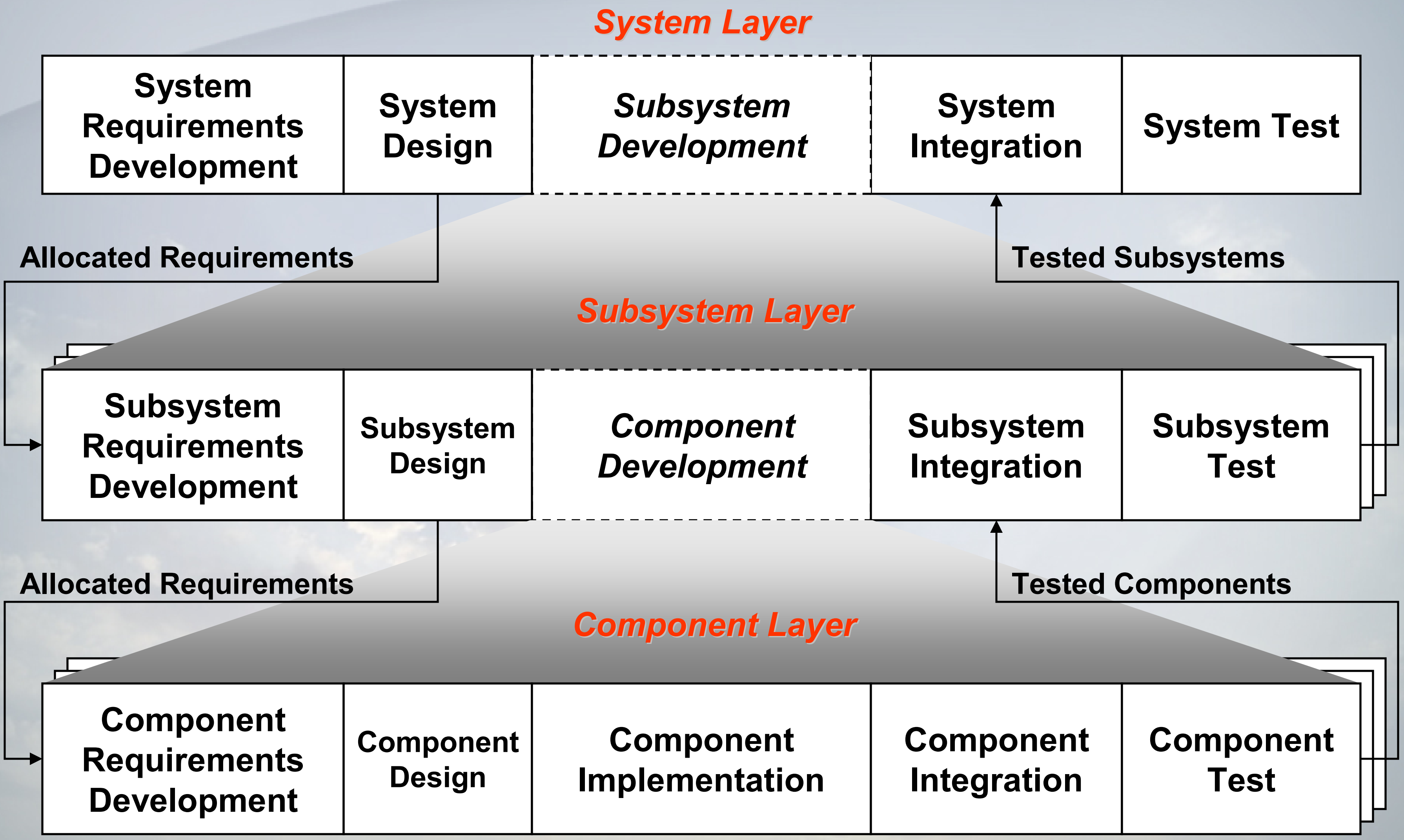
This architecture should define the levels of system elements to be developed…

- System of Systems
- Systems
- Segments
- Subsystems
- Software Items
- Assemblies
- Components
- Parts



System Development Layers

The following examples will use this simple 3-layer architecture:



Work Breakdown Structure

The project's work breakdown structure (WBS) should be developed in conjunction with the life-cycle planning activities.

- It is assumed that the customer has provided a high-level WBS (e.g., Contract WBS or CWBS) that defines the high-level deliverables tied to contract line items (CLINs).
- It is the project's responsibility to add needed detail to this WBS to fully develop the Program or Project WBS (PWBS) that will be the basis for detailed project planning and tracking.

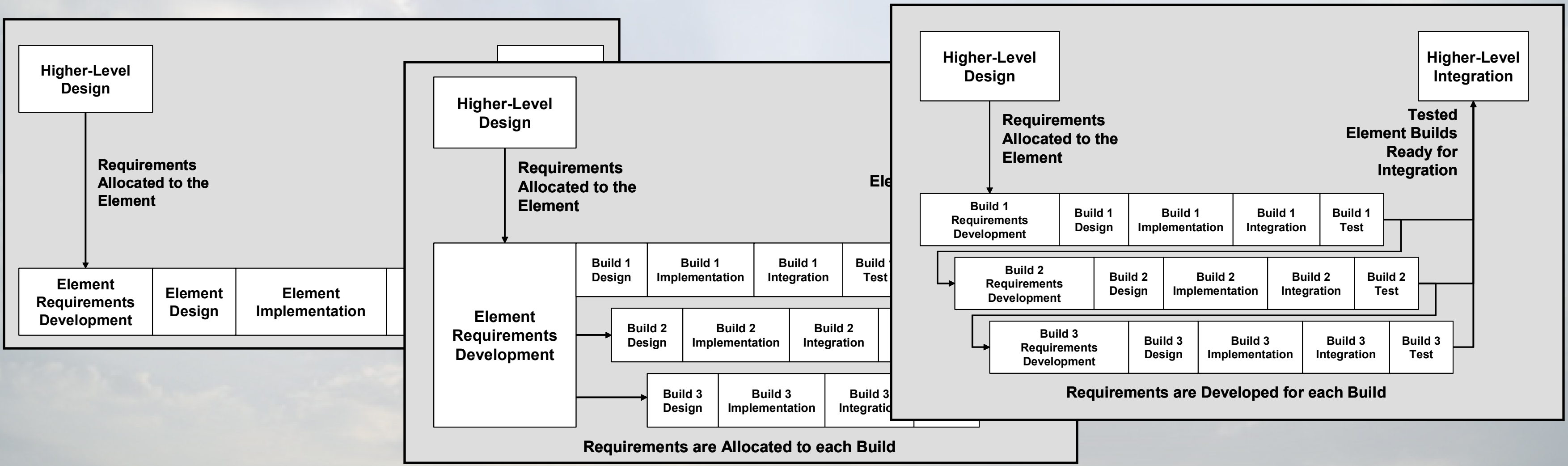
Life-cycle-based planning provides a mechanism for evolving the WBS to the appropriate level of detail in a systematic manner.

2. Identify Life-Cycle Strategy

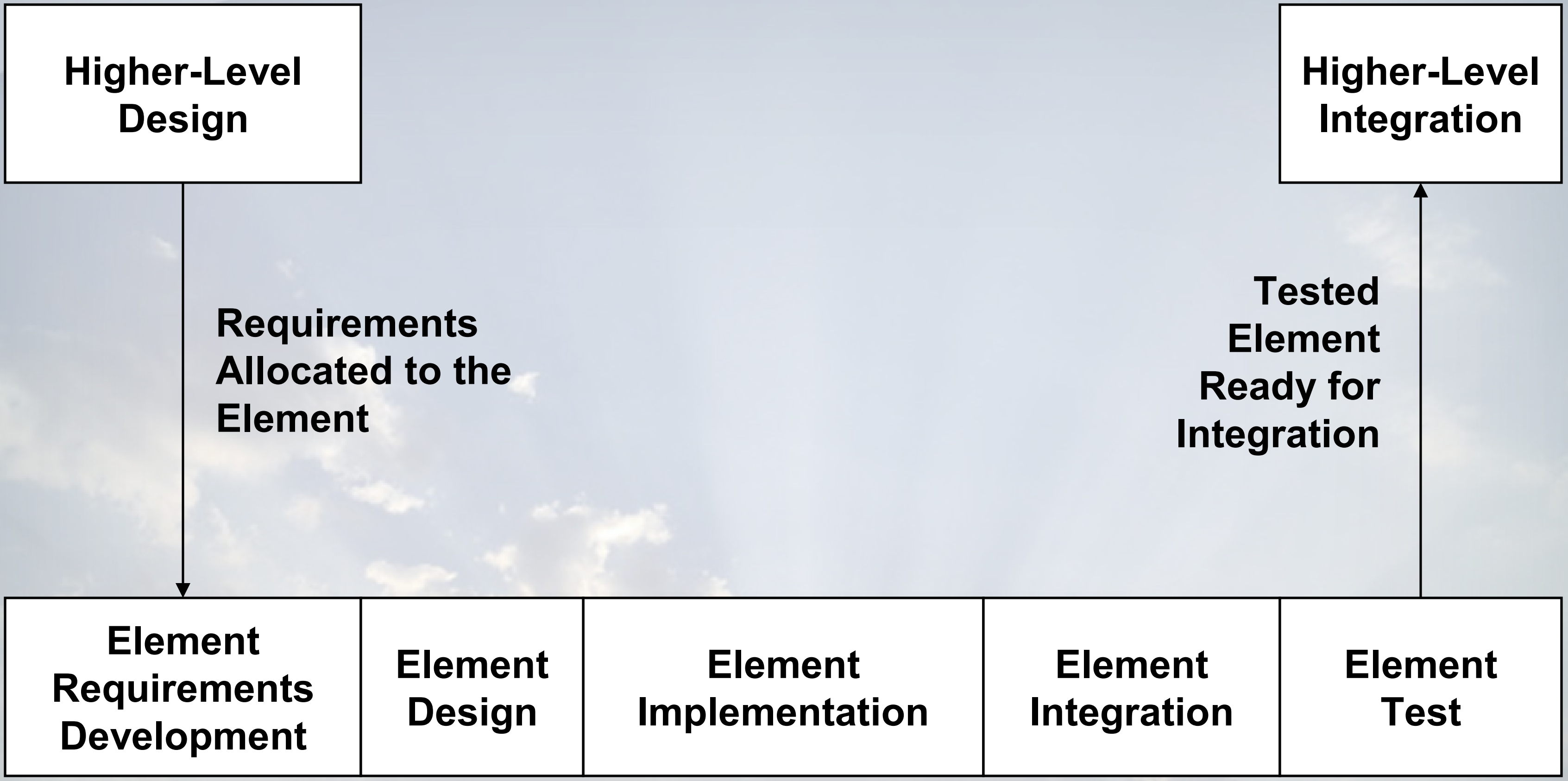
At each level of system development identify the life-cycle strategy to be followed:

- Once-through (Waterfall)
- Incremental
- Evolutionary (Spiral)

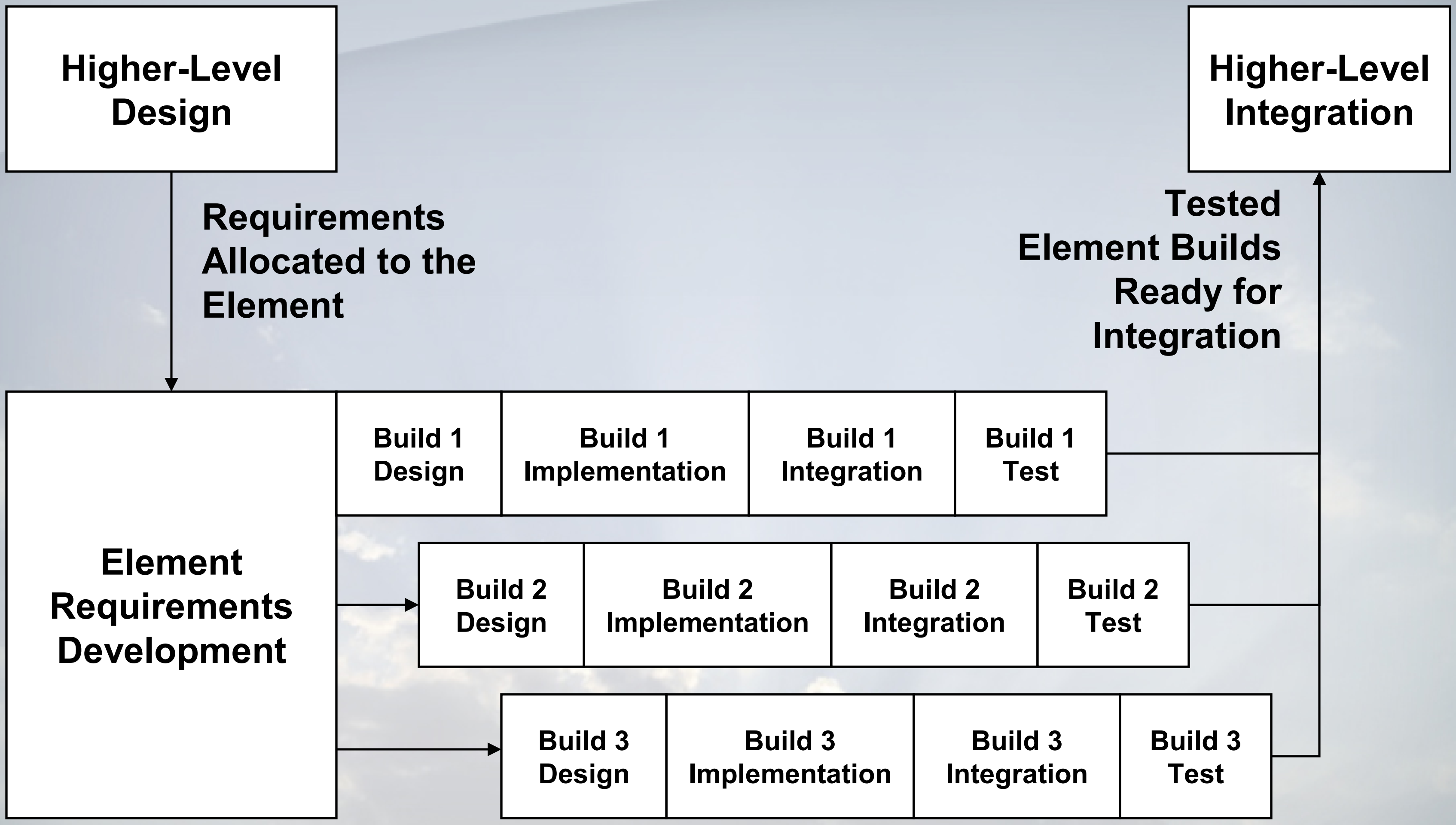
The project may employ multiple strategies for different system developments, depending upon the complexities, risks, and delivery requirements for each system element.



Once-Through Life-Cycle Strategy

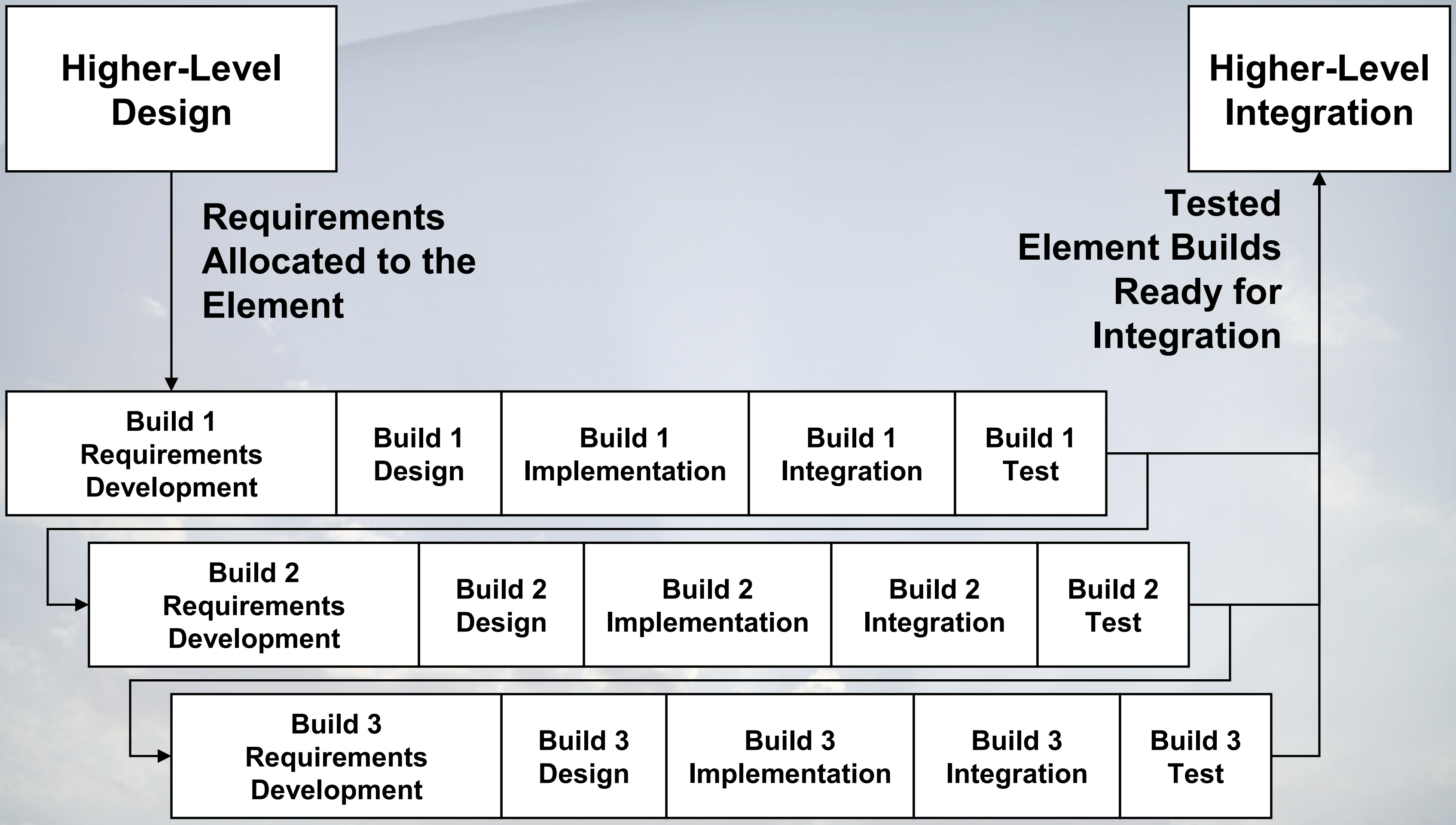


Incremental Life-Cycle Strategy



Requirements are Allocated to each Build

Evolutionary Life-Cycle Strategy



Requirements are Developed for each Build

Example - Overall System Strategy

- For the overall system, the customer wants Initial Operational Capability (IOC) in two years, and Final Operational Capability (FOC) in five years.
- IOC has some minimum requirements, but they would like as much capability as can be achieved with medium risk.
- FOC also has some minimum requirements, but the customer would like us to attempt to include some innovative capabilities.

These needs dictate an Evolutionary strategy at the system level with at least two iterations.

Example – Subsystem Strategies

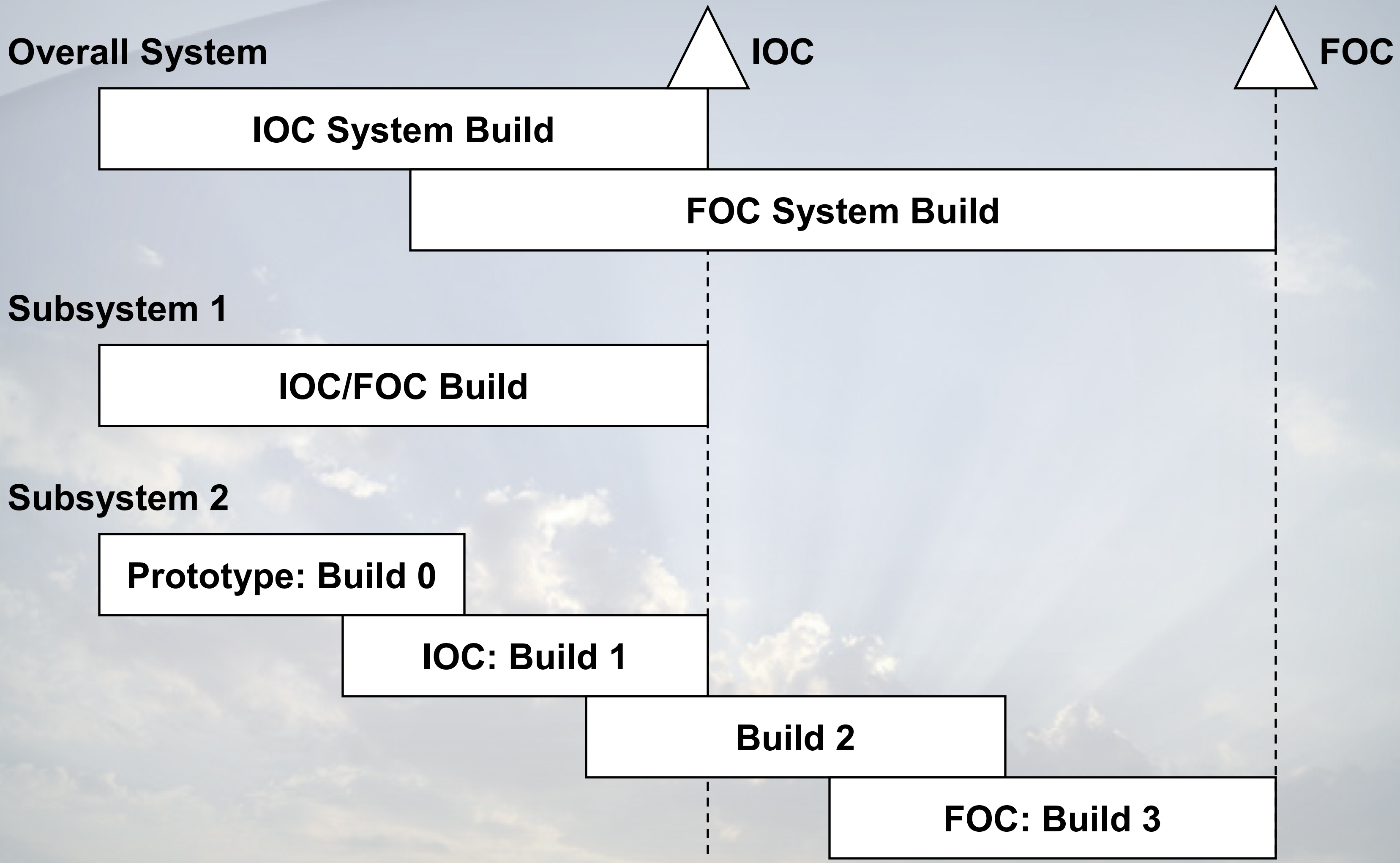
- One of the major subsystems is an upgrade to an existing legacy subsystem.
- The requirements are well defined and it is needed at full capability for IOC. It will not change for FOC.

The Once-Through (Waterfall) strategy is appropriate for this subsystem.

- Another major subsystem will employ an innovative networking technology that will require prototyping.
- The performance limits of this technology are not yet known.
- IOC requires this subsystem to be functional, but FOC requires a high performance level.

This subsystem will follow an Evolutionary strategy.

Example Build Plan



3. Define the Life-Cycle Phases

- For each development, identify the life-cycle phases to be performed in each iteration.
- If an element includes hardware and software development, identify the phases and iterations needed for each.
- Identify procurement and supplier management phases, as appropriate.

- Typical development life-cycle phases include:
  - Requirements Development
  - Preliminary Design
  - Detailed Design
  - Implementation
  - Integration
  - Testing (V&V)

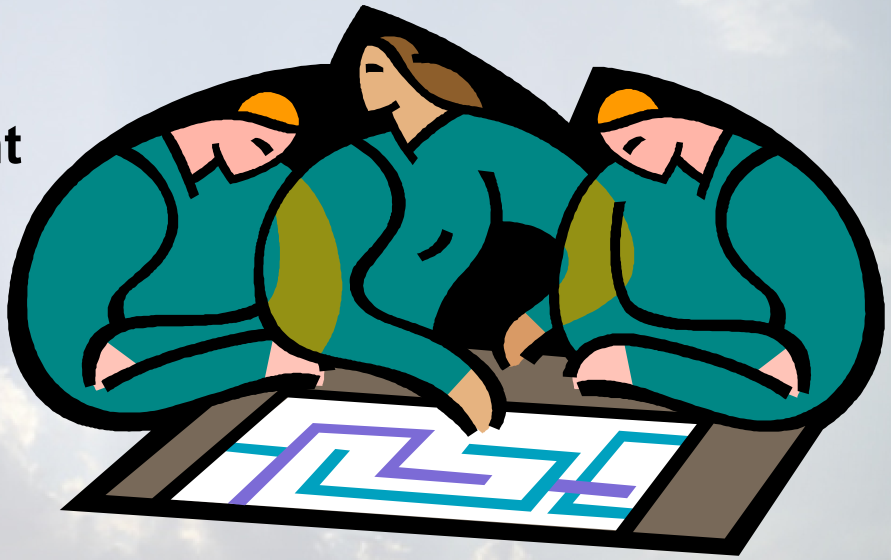
About Life-Cycle Phases

Phases may be defined at any or all system levels:

- System of Systems, System, Subsystem, etc.

The following are activities to be performed within phases, but are probably not themselves phases:

- Requirements Analysis
- Requirements Validation
- Requirements Management
- Verification
- Validation
- Qualification Testing
- Rework
- Product Release
- Project Planning (?)



Example System Level Phases

At the overall system level, the following phases may be defined for the IOC Build:

- System Requirements Development
- Preliminary System Design
- System Integration
- System Test

Subphases may be defined within phases to provide more definition; e.g.,

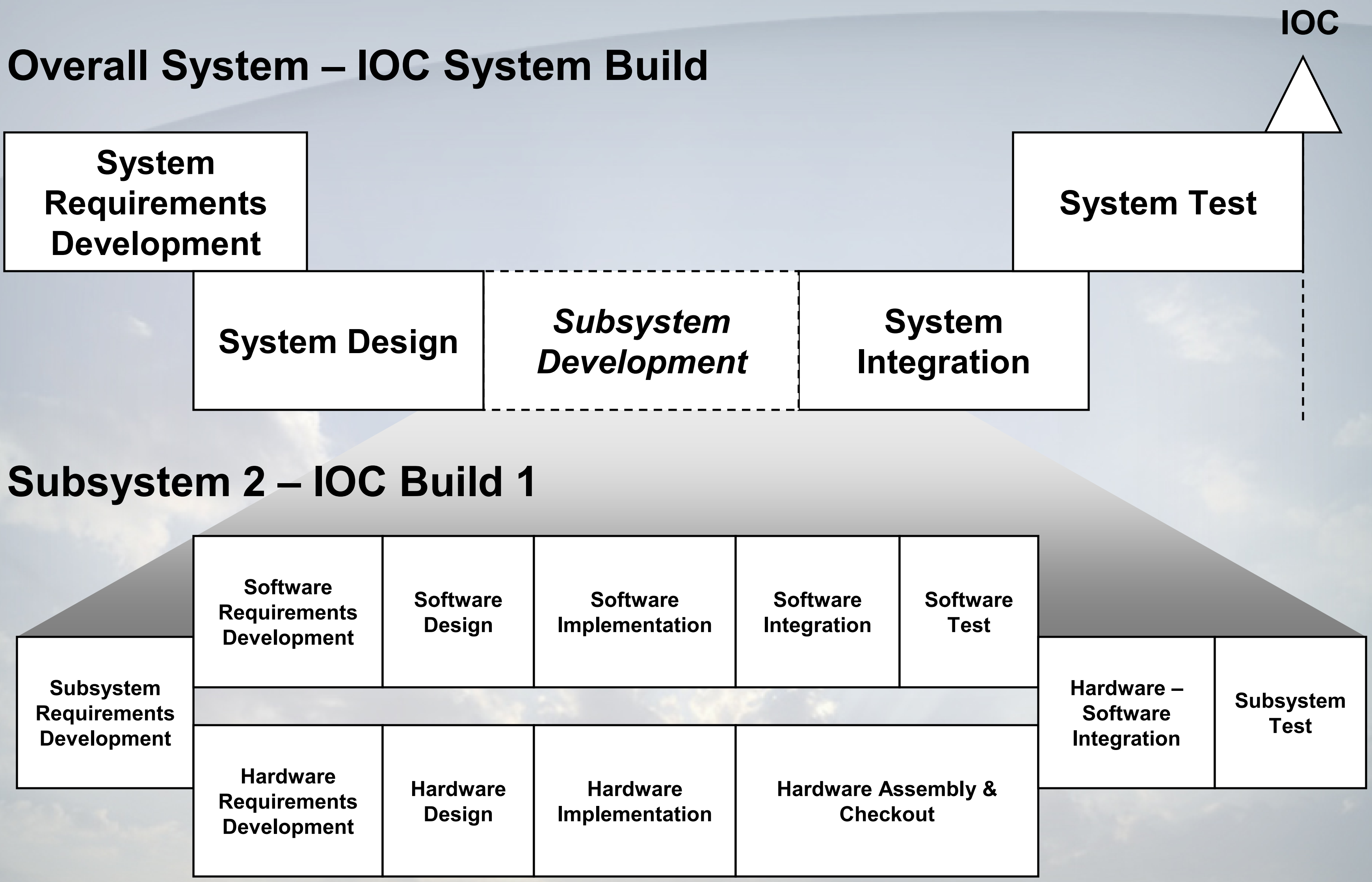
- System Test may be divided into…
  - Qualification Test
  - Flight Test
  - Operational Test (OT&E)

Example Subsystem Phases

At the subsystem level, Subsystem 2 may define the following phases for the IOC Build 1 iteration:

- Subsystem Requirements Development
  - Software Requirements Development
  - Software Design
  - Software Implementation
  - Software Integration
  - Software Test
  - Hardware Requirements Development
  - Hardware Design
  - Hardware Implementation
  - Hardware Assembly and Checkout
- Hardware-Software Integration
- Subsystem Test

Example Phase Relationships


IOC
Overall System – IOC System Build
System Requirements Development
System Test
System Design
Subsystem Development
System Integration
Subsystem 2 – IOC Build 1
Subsystem Requirements Development
Software Requirements Development
Software Design
Software Implementation
Software Integration
Software Test
Hardware Requirements Development
Hardware Design
Hardware Implementation
Hardware Assembly & Checkout
Hardware – Software Integration
Subsystem Test

4. Identify Iterations

If not done already, define the iterations required for each incremental and evolutionary strategy.

Considerations:

- Complexity
- Risk
- Maturity
- Delivery Needs
- Dependencies
- Constraints

Each iteration should have a well-defined goal.

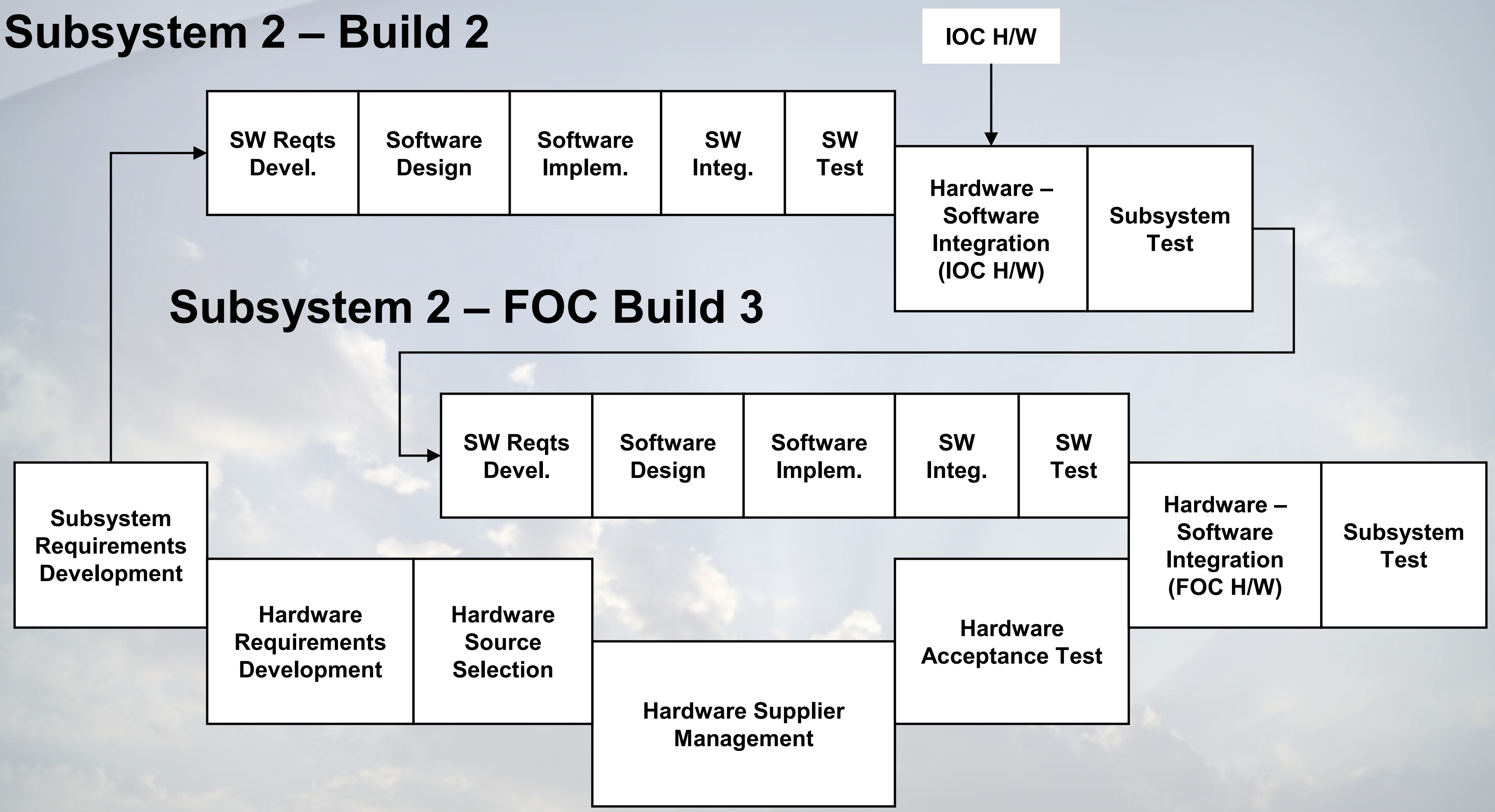
Example Iteration Goal

- Subsystem 2 FOC hardware will not be ready until 1 year before the FOC milestone.
- To reduce risk, the customer would like to validate some communication functions in an operational environment using IOC hardware.
- Subsystem 2 Build 2 will accommodate this validation by providing a software-only build 1 year into FOC development.

Iteration Goal:

Validate the FOC Communication Software using the IOC Hardware Configuration

Example Subsystem Iterations

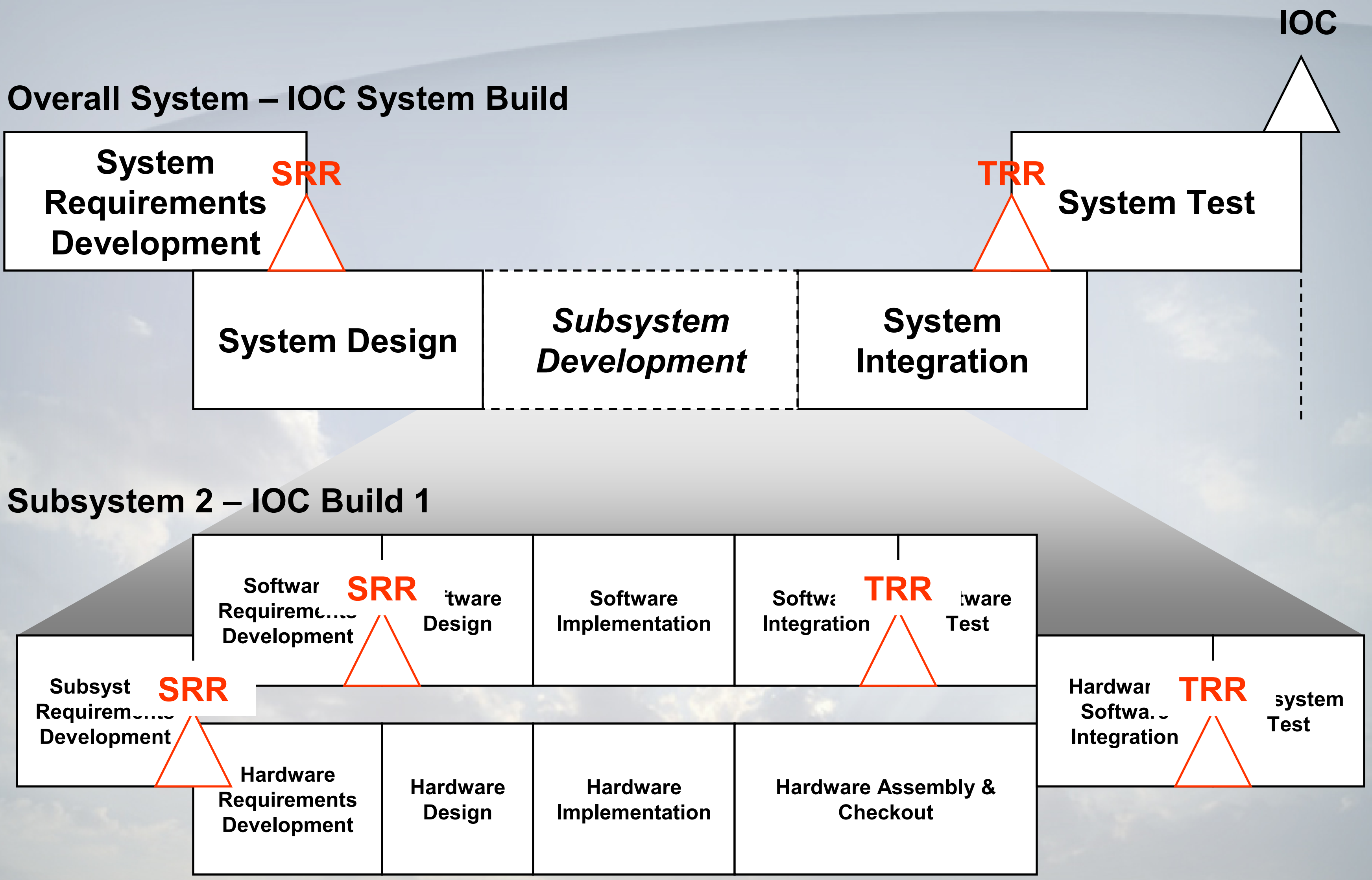

Subsystem 2 – Build 2
IOC H/W
SW Reqts Devel.
Software Design
Software Implem.
SW Integ.
SW Test
Hardware – Software Integration (IOC H/W)
Subsystem Test
Subsystem 2 – FOC Build 3
SW Reqts Devel.
Software Design
Software Implem.
SW Integ.
SW Test
Hardware – Software Integration (FOC H/W)
Subsystem Test
Subsystem Requirements Development
Hardware Requirements Development
Hardware Source Selection
Hardware Supplier Management
Hardware Acceptance Test

5. Identify Allocation and Integration Events

Define the key points within each level of system decomposition when requirements will be allocated to lower tier elements and when lower-tier elements will be integrated into subsystems and systems.

These are good points at which to schedule milestone reviews such as:
Subsystem Requirements Reviews (SRRs) and Test Readiness Reviews (TRRs).

Example Milestone Events


IOC
Overall System – IOC System Build
System Requirements Development
SRR
TRR
System Test
System Design
Subsystem Development
System Integration
Subsystem 2 – IOC Build 1
Software Requirements Development
SRR
Software Design
Software Implementation
Software Integration
TRR
Software Test
Subsystem Requirements Development
SRR
Hardware Software Integration
TRR
Subsystem Test
Hardware Requirements Development
Hardware Design
Hardware Implementation
Hardware Assembly & Checkout

Allocation and Integration Events

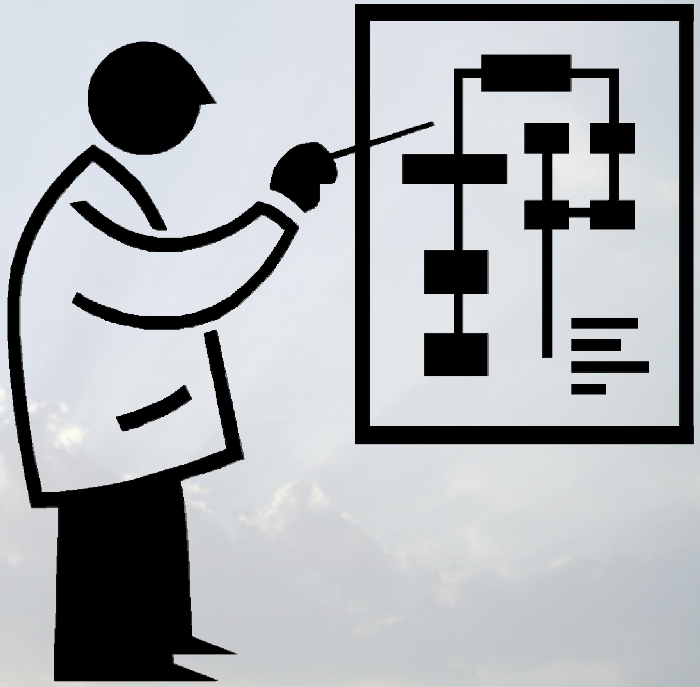
When scheduling the project, these events are the key points for defining dependencies.

- Subsystem development depends upon allocation of system-level requirements
- System-level testing depends upon completing the integration of subsystems

6. Identify Processes and Work Products

For each phase identified within each system element's life cycle:

- Identify the processes that will be used to perform the activities in that phase
- Identify the major work products to be produced or updated during that phase.



Example Phase Processes and Work Products

For Subsystem 2:

Phase	Processes	Work Products
Subsystem Requirements Development	• Standard Project Planning Process • Standard Requirements Development Process • Standard Requirements Management Process	• Project Plan • Requirements Specification • Technical Performance Measures • Functional Architecture • Operational Concepts and Scenarios • Requirements Traceability

7. Define the Project Plans and Schedule

Once the project's life-cycle phases have been defined in terms of processes and work products, define the overall project plans and schedules.

- A typical way to represent the overall planning with dependencies is with an Integrated Master Plan (IMP) and Integrated Master Schedule (IMS).
- However the overall plan is documented, it should be consistent with and/or traceable to the project's WBS.

8. Develop Tasks, Estimates, Staffing

Using the overall project plans and schedules, define tasks that can be used as the basis for detailed estimating and staffing.

- **The tasks can be defined in terms of work packages which will be the basis for earned-value management.**
- **Again, the tasks and work packages should be consistent with the project's WBS.**

Integration Considerations

- The steps described here may be performed iteratively or recursively, until the entire project scope has been defined.
- Life-cycle planning should be tightly integrated with project planning, requirements development, and process set definition.
- Multiple levels of life-cycle descriptions may be developed all at once or each level may be developed as its corresponding system element is defined.

Conclusions

- Development life-cycles should be defined at all levels of the system architecture
- Life-cycle strategies, phases, and iterations should be defined as part of project planning
- Engineering processes should be mapped to all phases of system development
- Allocation and integration events define milestones and dependencies in the project's plans

The Bottom Line:
Use of a life-cycle planning process should result in project plans that are more accurate, comprehensive, and lower risk!