IEA-P – DEPARTAMENTO DE PROJETOS
(PROJECT DEPARTMENT)

SYSTEMS ENGINEERING REVIEW

Prepared by Prof. Christopher S. Cerqueira

[SIS-08][2025]

Summary

Context

At the beginning there was only chaos

Chaos (cosmogony) - Wikipedia

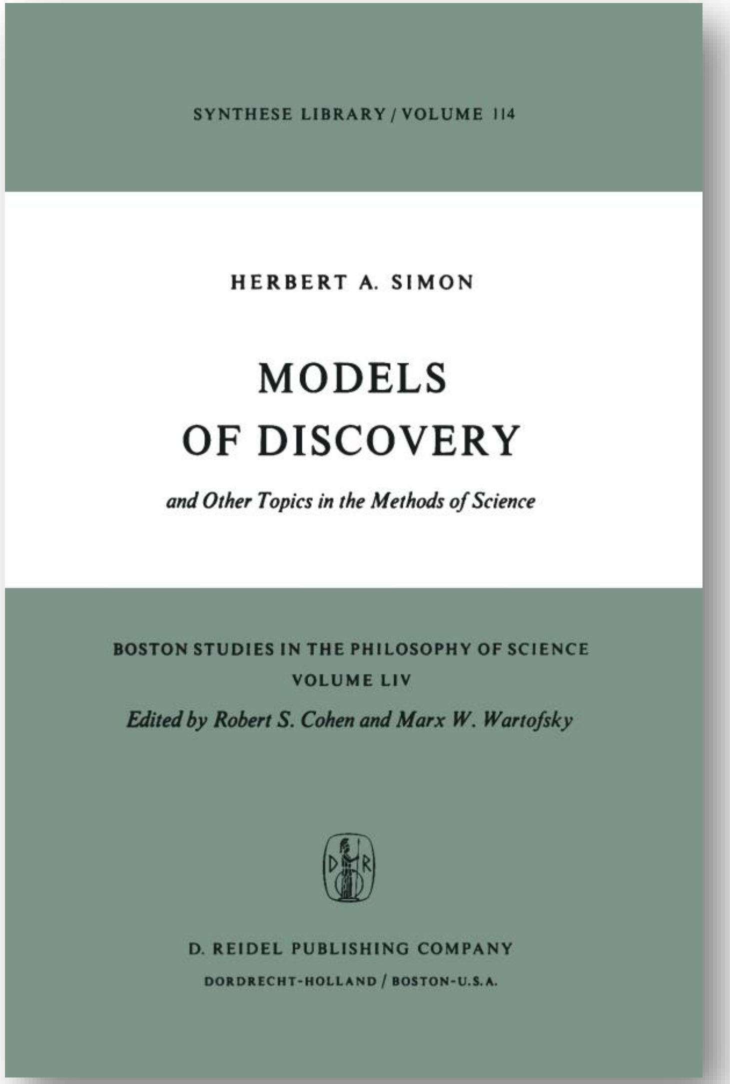
- **At the beginning there was only Chaos, Night, dark Erebus, and deep Tartarus. Earth, the air and heaven had no existence.** [695] Firstly, blackwinged Night laid a germless egg in the bosom of the infinite deeps of Erebus, and from this, after the revolution of long ages, sprang the graceful Eros with his glittering golden wings, swift as the whirlwinds of the tempest. He mated in deep Tartarus with dark Chaos, winged like himself, and thus hatched forth our race, which was the first to see the light. [700] That of the Immortals did not exist until Eros had brought together all the ingredients of the world, and from their marriage Heaven, Ocean, Earth and the imperishable race of blessed gods sprang into being. Thus our origin is very much older than that of the dwellers in Olympus. We are the offspring of Eros; there are a thousand proofs to show it. We have wings and we lend assistance to lovers. [705] How many handsome youths, who had sworn to remain insensible, have opened their thighs because of our power and have yielded themselves to their lovers when almost at the end of their youth, being led away by the gift of a quail, a waterfowl, a goose, or a cock.

Aristophanes, Birds, line 685 (tufts.edu)

We started to understand the chaos

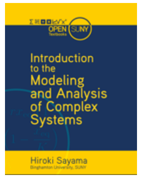
Figure 1 – The Flammarion woodcut (19th Century), illustrating the Flat-Earth cosmology. Seen from the observer's village, the Earth seems flat, as encountered in everyday experience. However, just to the left, a "curious" fellow decides to breach the sphere of the fixed stars to sneak a peek at the mechanisms that move the Sun, Moon and planets.

https://doi.org/10.1590/S0103-40142006000300022

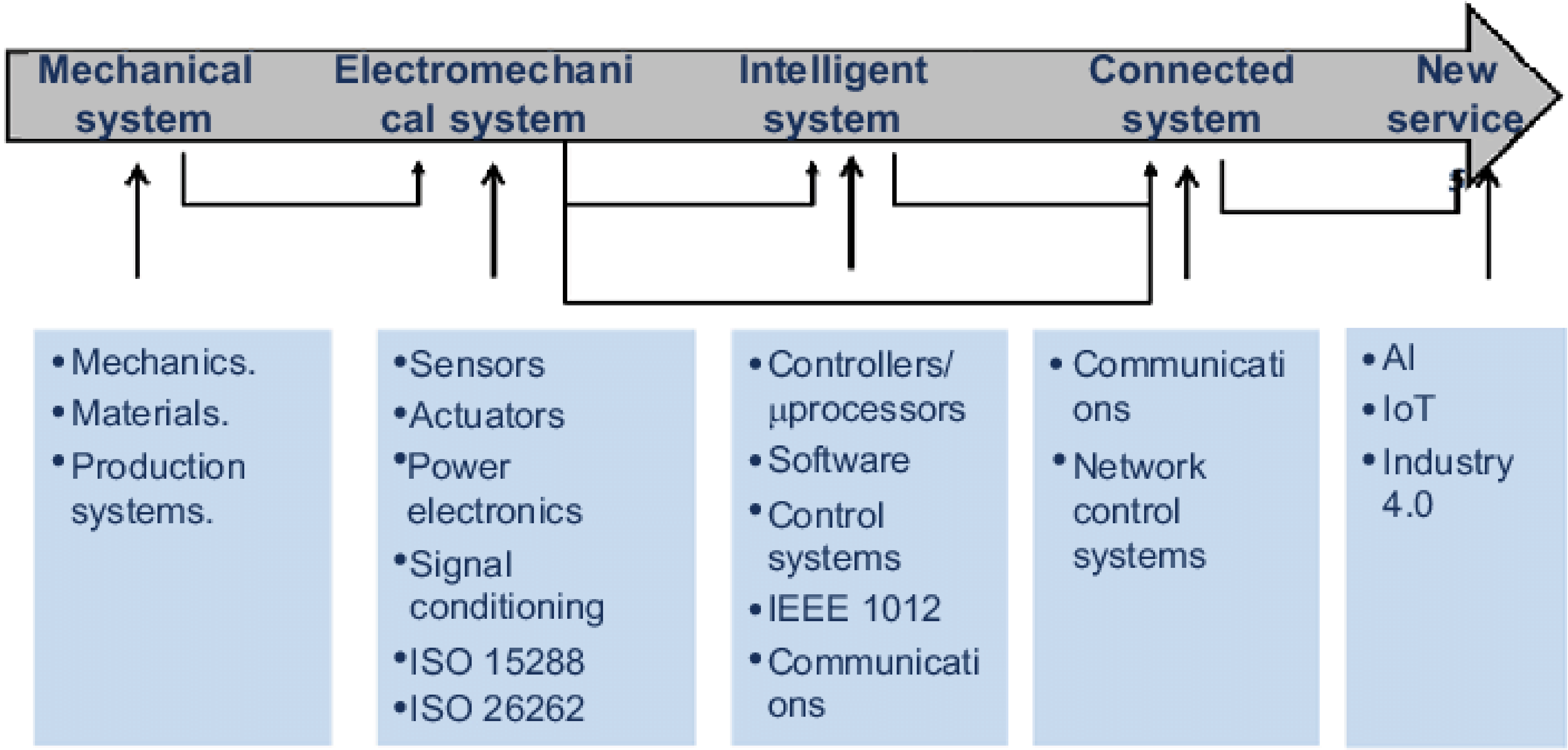


Complex systems

- **Complex systems** are **networks** made of a number of **components that interact with each other**, typically in a **nonlinear** fashion. Complex systems may arise and evolve through **self-organization**, such that they are neither completely regular nor completely random, permitting the development of **emergent behavior** at macroscopic scales.



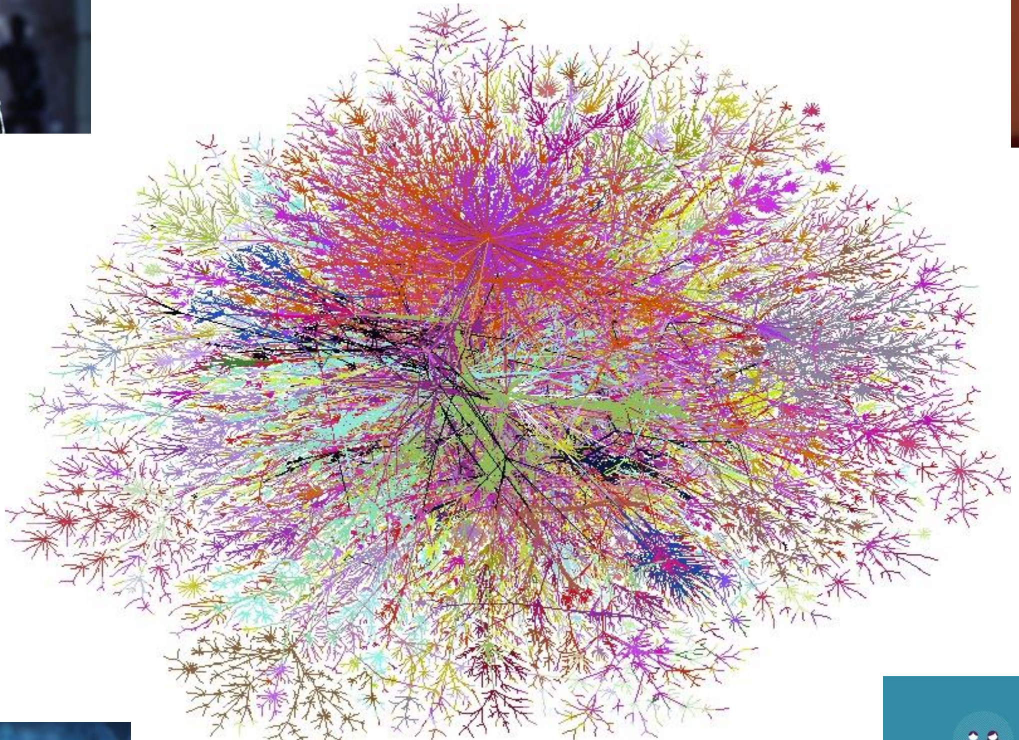
The growth of complexity in systems



https://doi.org/10.4271/12-03-02-0011

Ending scene from 1971 movie THX 1138

https://www.youtube.com/watch?v=atMdf0rhbpI



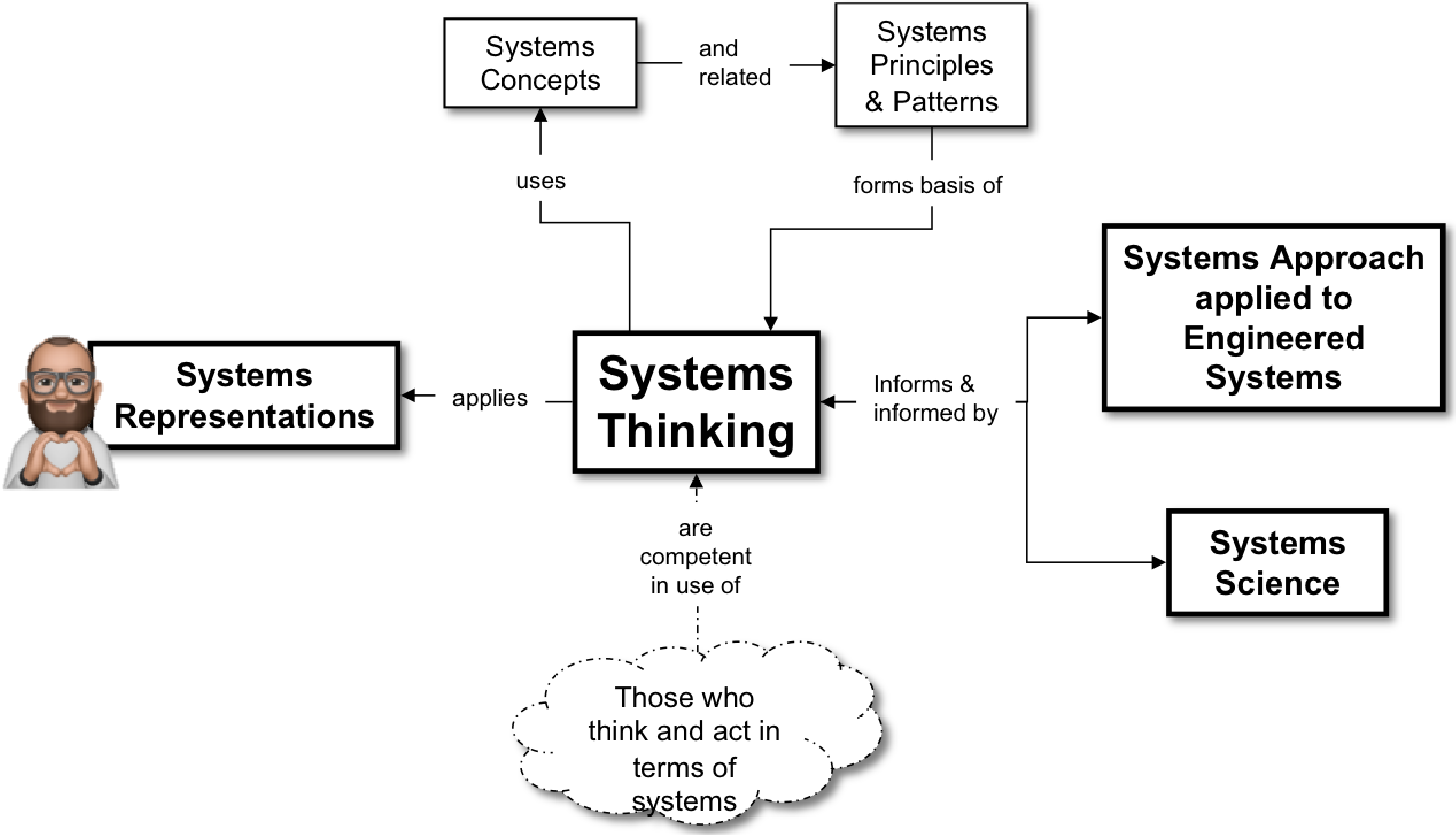
https://ioannouolga.blog/2017/09/14/complexity-theory-ii-m-woermann/

Some definitions

Systems thinking

[Concepts of Systems Thinking - SEBoK (sebokwiki.org)]

[Principles of Systems Thinking - SEBoK (sebokwiki.org)]



[Systems Thinking and Systems Science RA - Systems Thinking - SEBoK (sebokwiki.org)]

What is Systems Engineering?

Systems Engineering is **a transdisciplinary approach and means**, based on systems principles and concepts, to enable the successful realization, use and retiral of engineered systems.

It focuses on

- establishing stakeholders' **purpose and success criteria**, and defining actual or anticipated customer needs and required functionality early in the development cycle,
- establishing an **appropriate lifecycle model** and process approach considering the levels of complexity, uncertainty and change
- documenting and **modelling requirements** and **solution architecture** for each phase of the endeavour
- proceeding with **design synthesis and system validation**
- while considering the **complete problem** and **all necessary enabling systems and services**.

Systems Engineering provides facilitation, guidance and leadership to integrate all the disciplines and specialty groups into a team effort forming an appropriately structured development process that proceeds from concept to production to operation, evolution and eventual disposal.

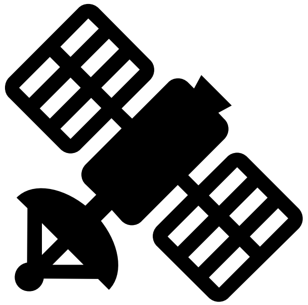
Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality solution that meets the needs of users and other stakeholders and is fit for the intended purpose in real-world operation, and avoids or minimizes adverse unintended consequences.

The practice of Systems Engineering

- The **practice** of systems engineering is concerned with both a **systemic** approach to understanding the problem, devising a solution, and understanding the interdependencies in the work to develop, deliver and evolve the solution;
- and a systematic approach to establishing objectives and success criteria, analyzing and documenting the solution, predicting its effectiveness, and establishing and implementing an effective and efficient process for development, delivery and subsequent evolution.

Engineered System x System Engineering

An engineered system is an system made of technical or sociotechnical elements that exhibits emergent properties not exhibited by its individual elements. It is created by and for people; has a purpose, with multiple views; satisfies key stakeholders' value propositions; has a life cycle and evolution dynamics; has a boundary and an external environment; and is part of a system-of-interest hierarchy.

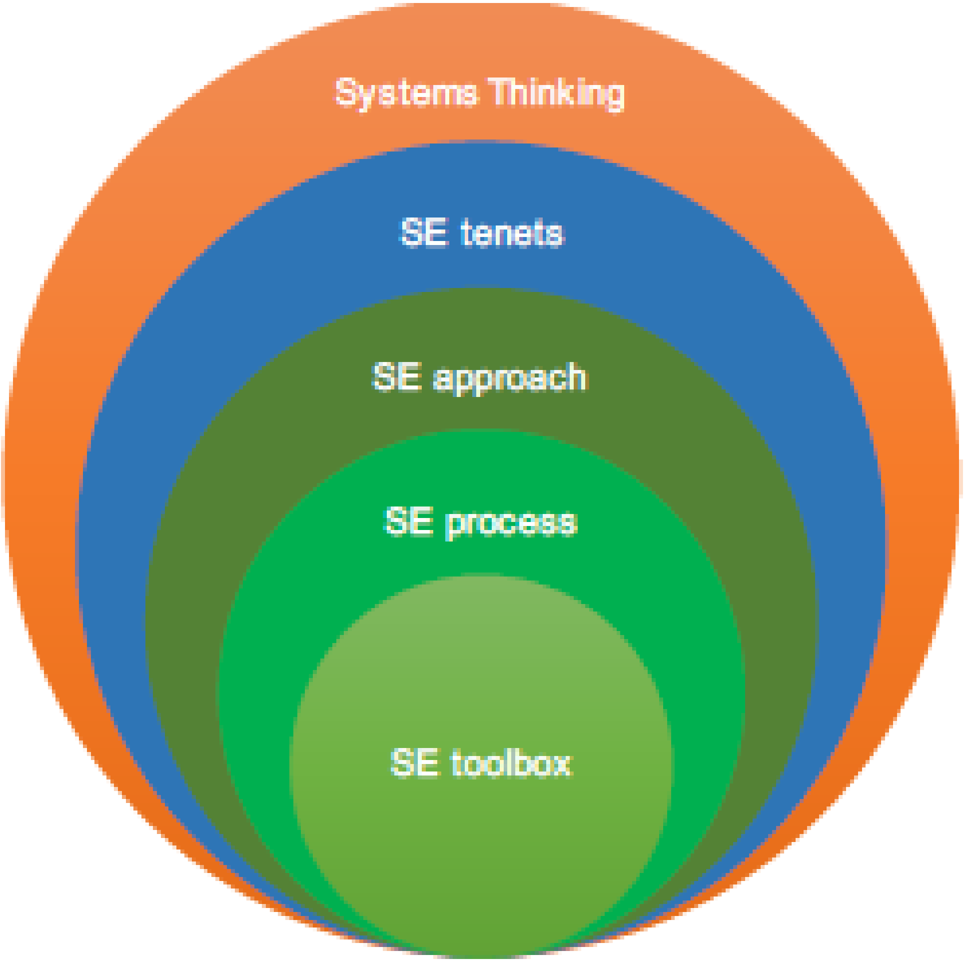


Systems engineering is "an transdisciplinary approach and means to enable the realization of successful (engineered) systems". It focuses on holistically and concurrently understanding stakeholder needs; exploring opportunities; documenting requirements; and synthesizing, verifying, validating, and evolving solutions while considering the complete problem, from system concept exploration through system disposal.

http://sebokwiki.org/wiki/Use_Case_0:_Systems_Engineering_Novices

Four aspects of Systems Engineering

- 1. some very basic and widely applicable **SE tenets** (principles and beliefs);
- 2. a general **SE approach** to complex and complicated problems;
- 3. the "**SE process**", which we see evolving from the current SE process described in the INCOSE SE Handbook into a family of SE process models, targeted towards different system types; and
- 4. an **SE toolbox** of techniques and methods that are widely applicable across the spectrum.



Tenants

- Systems Engineers often talk about using a "**systems approach**" to work their way into a problem that seems wider or fuzzier than "**normal engineering**" (whatever that is 🤷‍♂️).
- **Such a "systems approach" uses selected systems principles or beliefs that are of proven value in an engineering context and are also useful elsewhere.**
- We will use the term "**systems engineering tenets**" to refer to a key set of principles and beliefs drawn from various branches of systems thinking and the systems sciences, that seem to underpin most or all of what we currently recognize as systems engineering.

12 Systems Engineering Tenets

Envisioning Systems Engineering as a Transdisciplinary Venture (researchgate.net)

1. Understand what **success means**
2. Consider the **whole problem**, the **whole solution** and the **full lifecycle**
3. Understand and manage **interdependencies**
4. Adapt the parts to **serve the purpose** of the whole
5. Recognize that Systems Engineering occurs at **multiple levels**
6. Base decisions on **evidence and reasoned** judgement
7. **Recognize uncertainty** while managing change, risk opportunities and expectations
8. Handle **structure and behavior** as two complementary aspects of any system
9. Understand and use appropriate **feedback (loop)**
10. Understand and manage **value**
11. Be both **systemic and systematic**
12. **Respect** the people

Principle of Emergence:

<u>As the entities of a system are brought together, their interaction will cause function, behavior, performance, and other intrinsic properties to emerge.</u>

- Emergence is the **power and the magic of systems**. Emergence refers to **what appears, materializes, or surfaces when a system operates**. Obtaining the desired emergence is why we build systems. **Understanding emergence is the goal**—and the art—of system thinking.
- What emerges when a system comes together? **Most obviously and crucially, function emerges.** Function is what a system does: its actions, outcomes, or outputs. In a designed system, we design so that the anticipated desirable primary function emerges (cars transport people).

TABLE 2.1 | Types of emergent functions

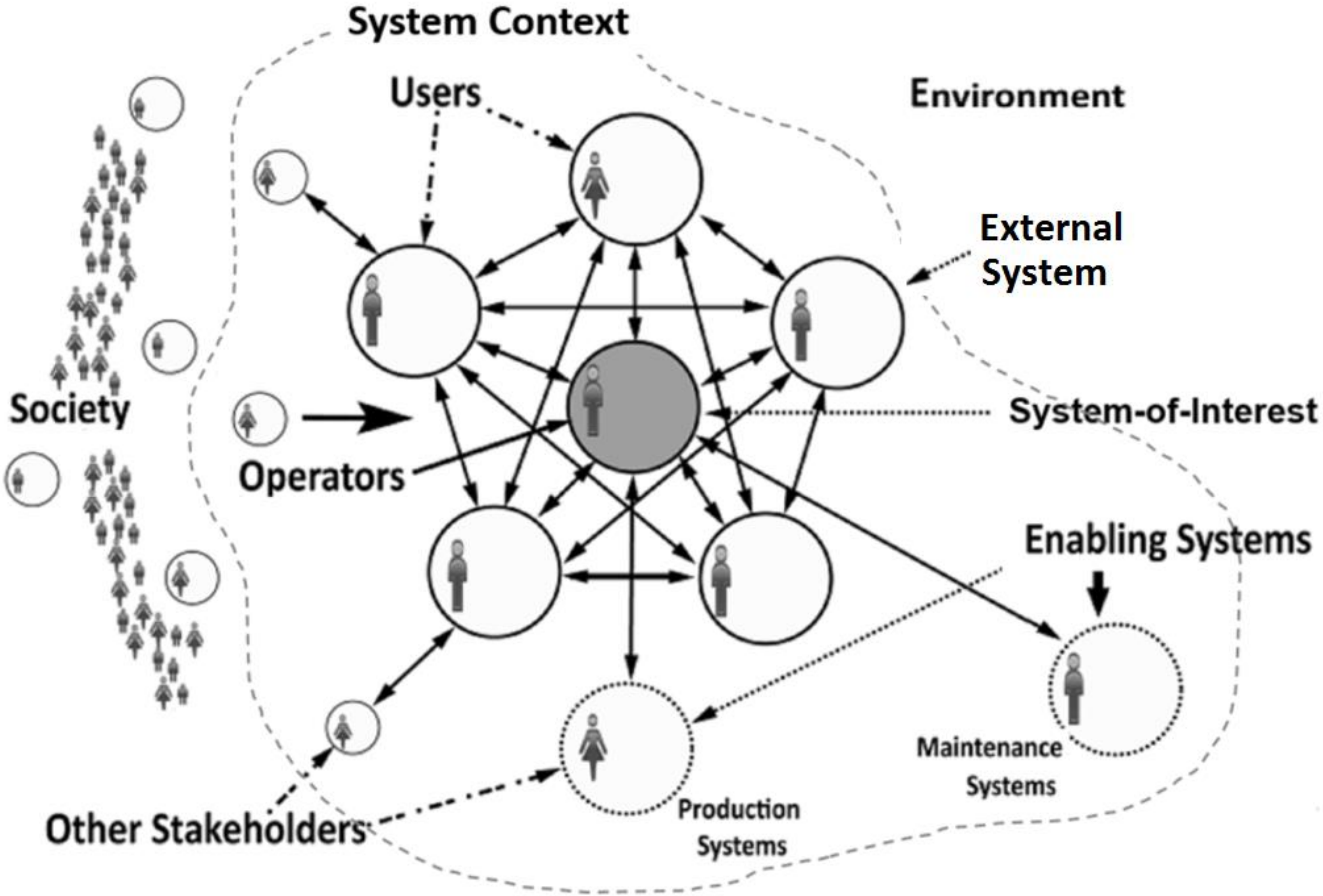
	Anticipated Emergence	Unanticipated Emergence
Desirable	Cars transport people Cars keep people warm/cool Cars entertain people	Cars create a sense of personal freedom in people
Undesirable	Cars burn hydrocarbons	Cars can kill people

Stakeholders

Stakeholders

- A stakeholder is any individual, group or organization that can affect, be affected by a project.

STAKEHOLDER DEFINITION


System Context
Users
Environment
External
System
Society
Operators
System-of-Interest
Enabling Systems
Maintenance
Systems
Production
Systems
Other Stakeholders

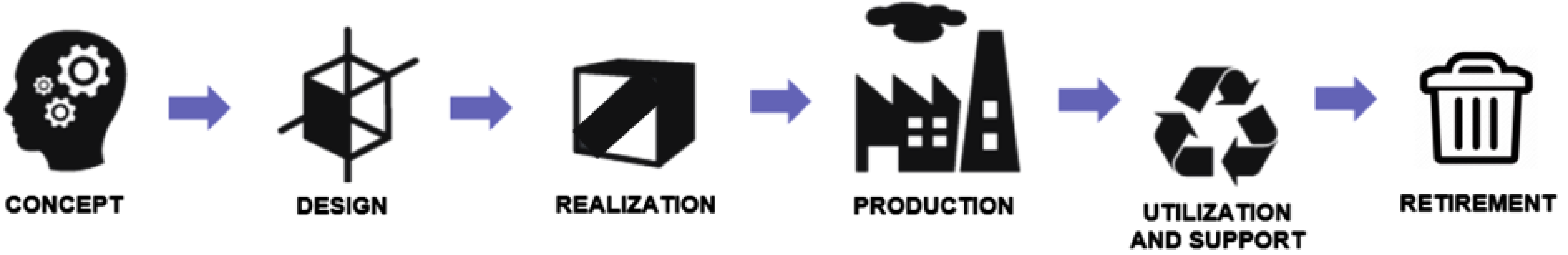
Life cycle

Life cycle

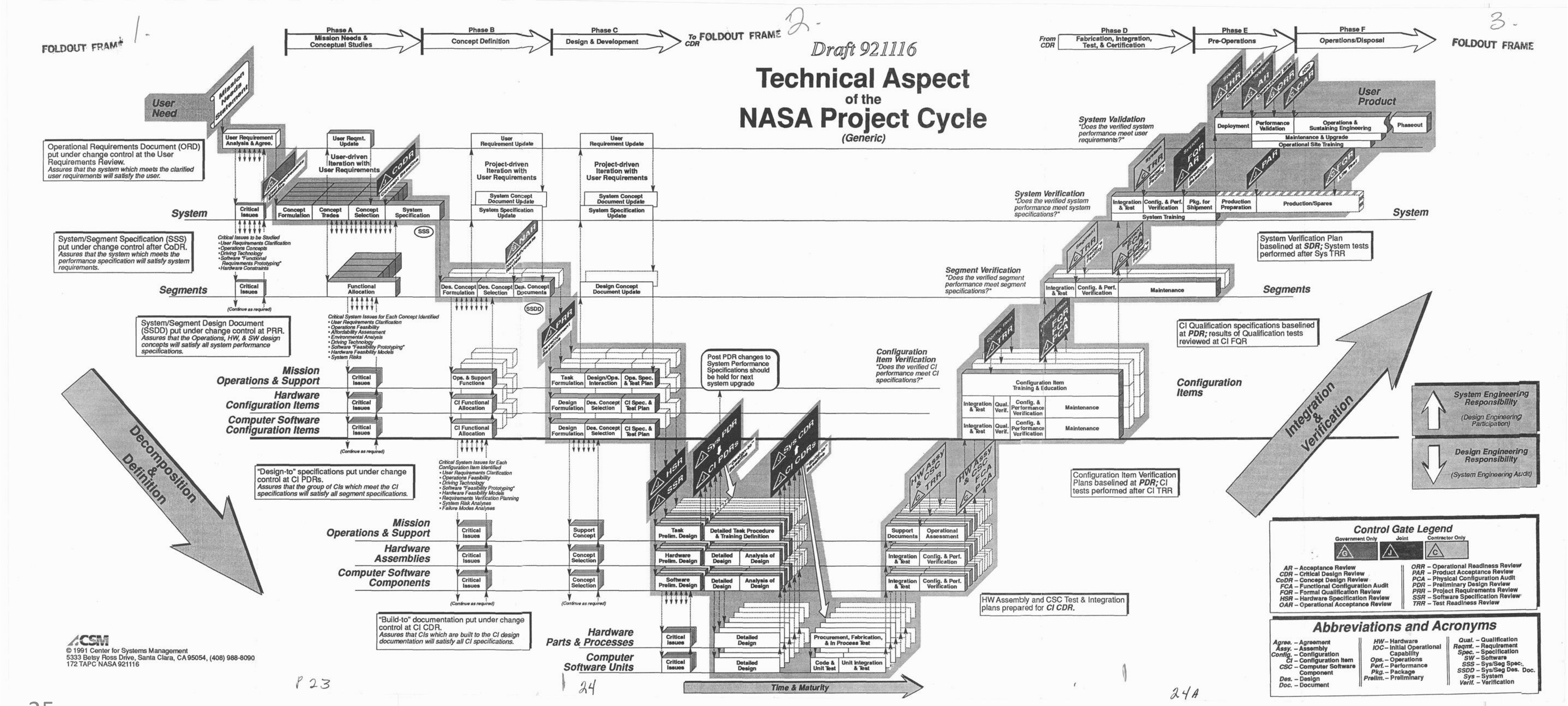
"Life cycle is the ***series of phases*** *through which* ***something*** *passes."*

Life cycle

- Engineered Systems have a **life cycle.**
  - Life cycle is a series of **stages through which a system passes** during its lifetime
  - Life cycle considers the evolution of a system **from conception through retirement**

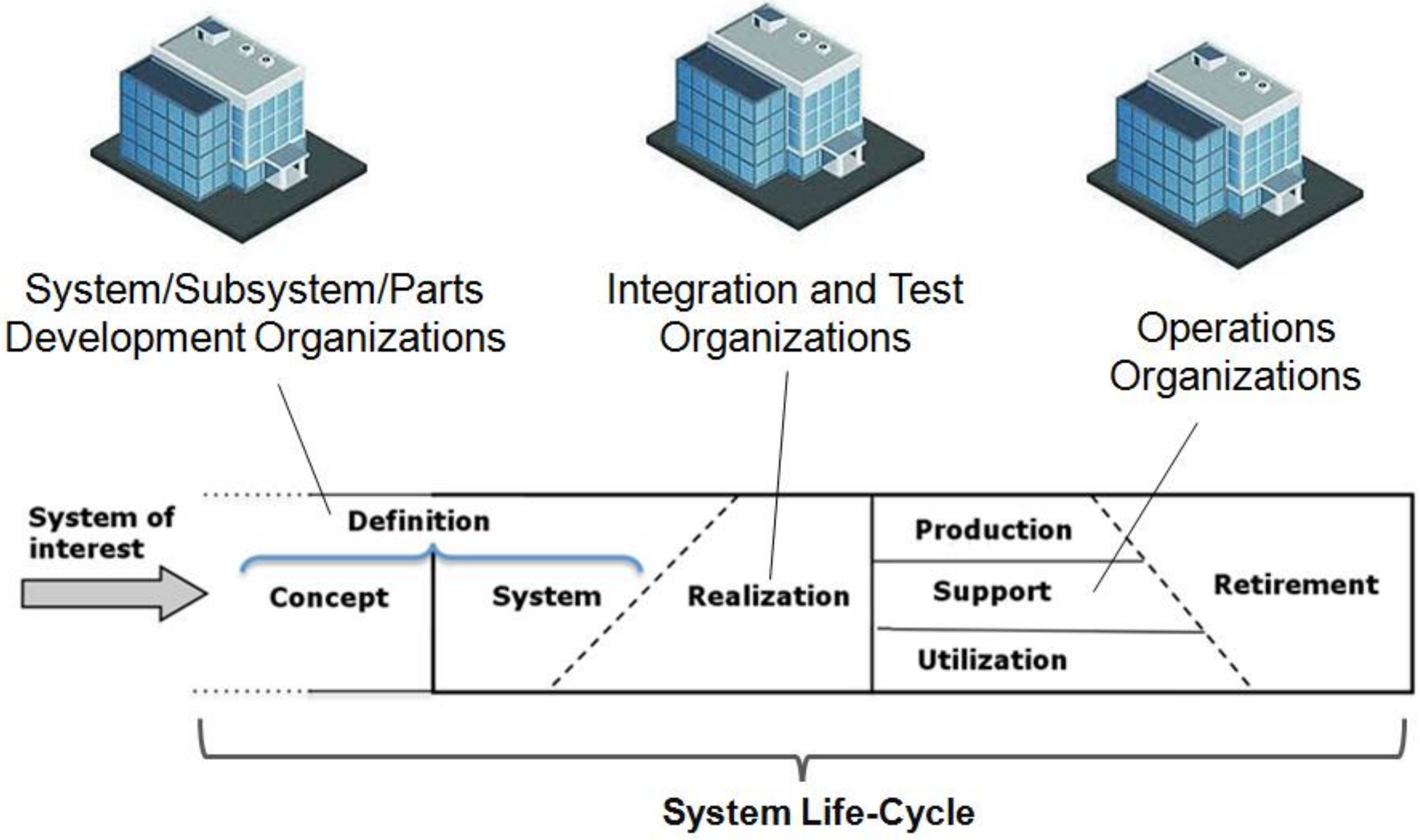


(Classical) VEE Model



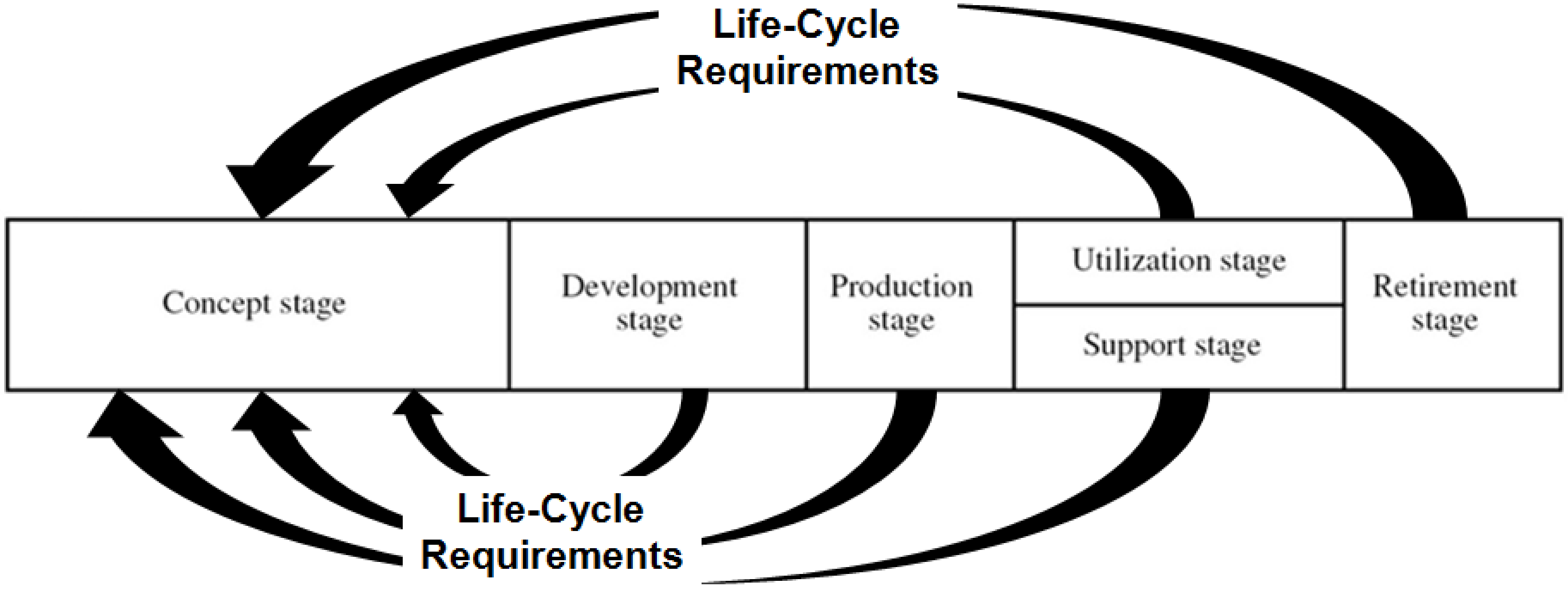
Life cycle concepts

- The life cycle concept is a **description** of the expected system life cycle.
- Life cycle concepts focus on **defining solutions** for the system life cycle.



Life cycle requirements

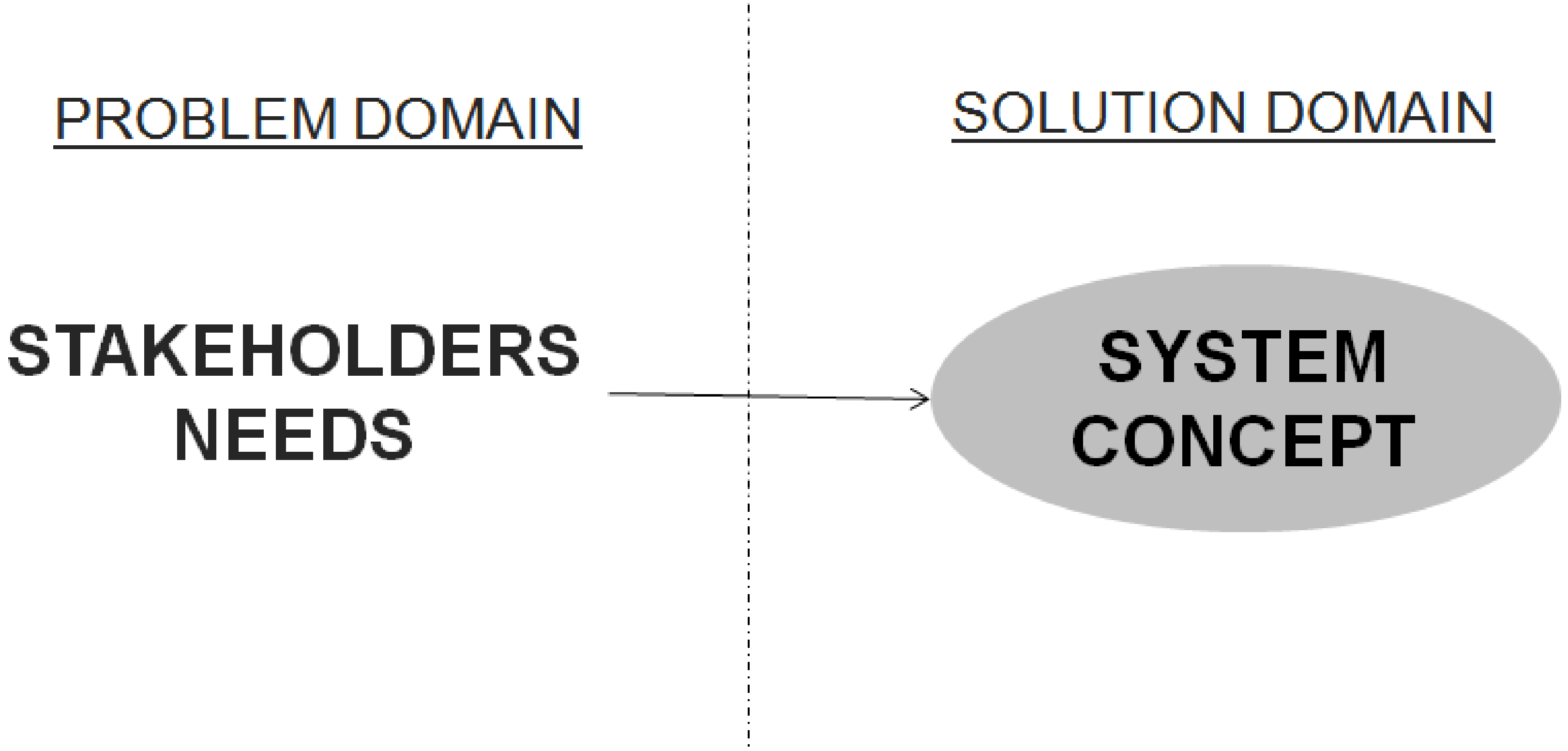
- Life cycle requirements promote the **anticipated understanding** about future system attributes



CONOPs

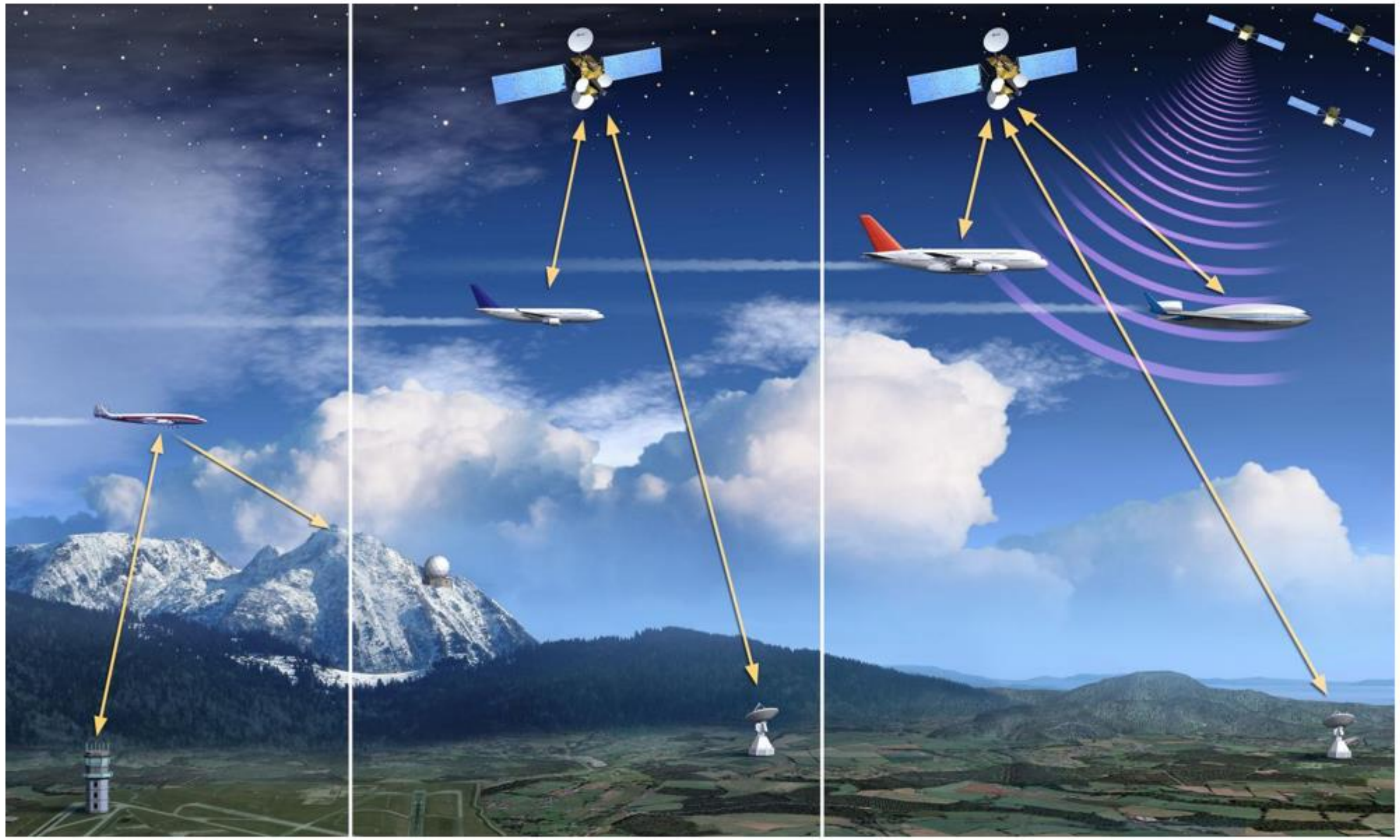
Concept definition phase

- The conceptual design provides a **description of the proposed system** that **fulfills the stakeholders needs**.



Concept of Operations

- Describes the **characteristics** of a proposed system from the viewpoint its operators



What is a CONOPS?

- Description of **how** the System will be operated to meet stakeholder expectations
- Explains your system's characteristics from an operational perspective and helps facilitate an understanding of the **system's purpose**
- Illustrates a day in the life of your system's **intended use**

Why is a CONOPS important?

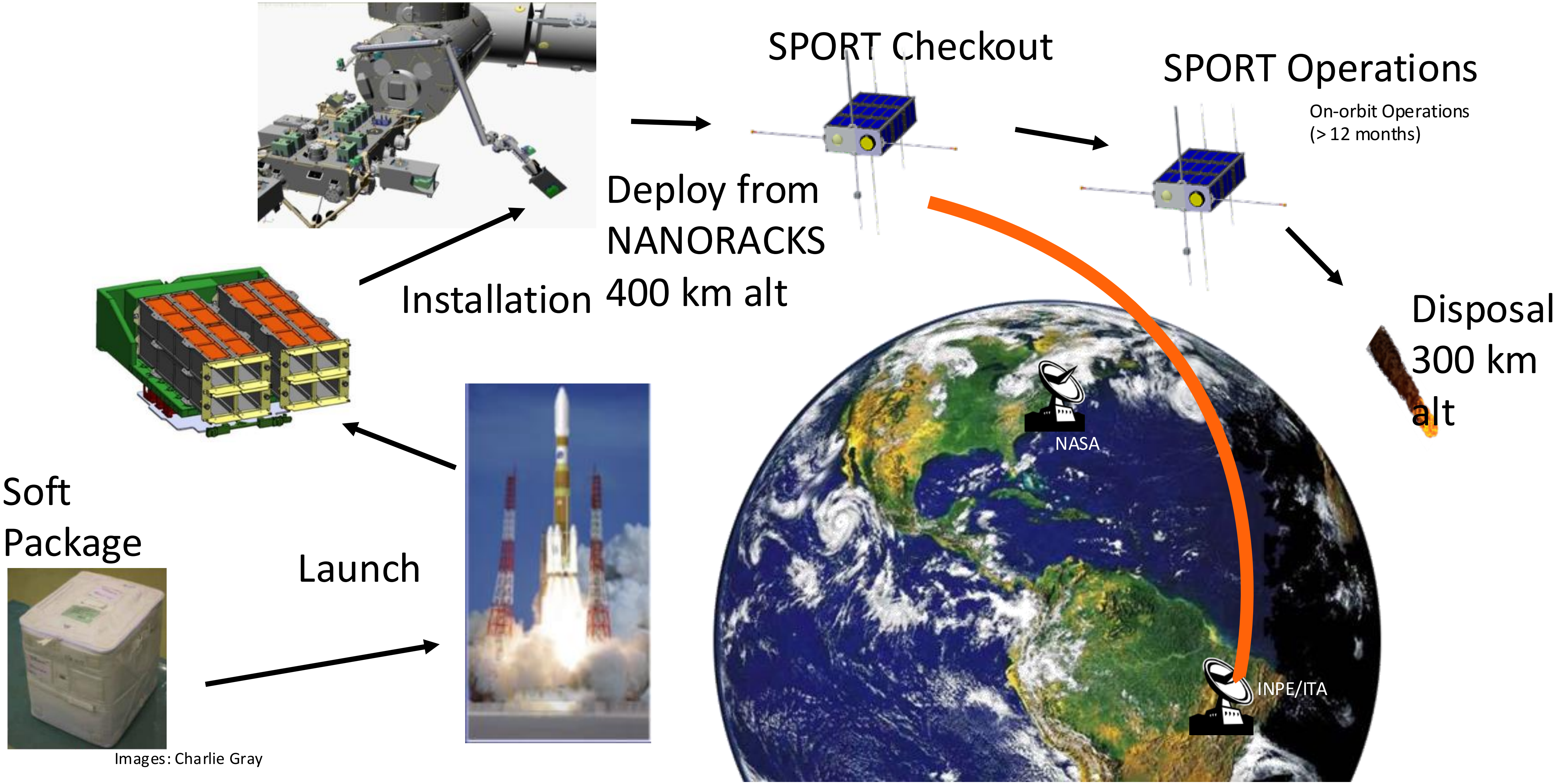
- Drives **development of requirements**
  - Maintains the **context** of a requirement in everyday, informal language
  - Thinking through the ConOps and use cases **reveal requirements and design functions** that might otherwise be overlooked
- Gets everyone on the same page about **what the project is and what it will do**
- Identifies **user interface** issues early
- Identifies **key stakeholder needs** for defining, designing, and implementing the end product
- Provides **guidance** for the development of system definition documentation

The CONOPS contains, as a minimum, the following:

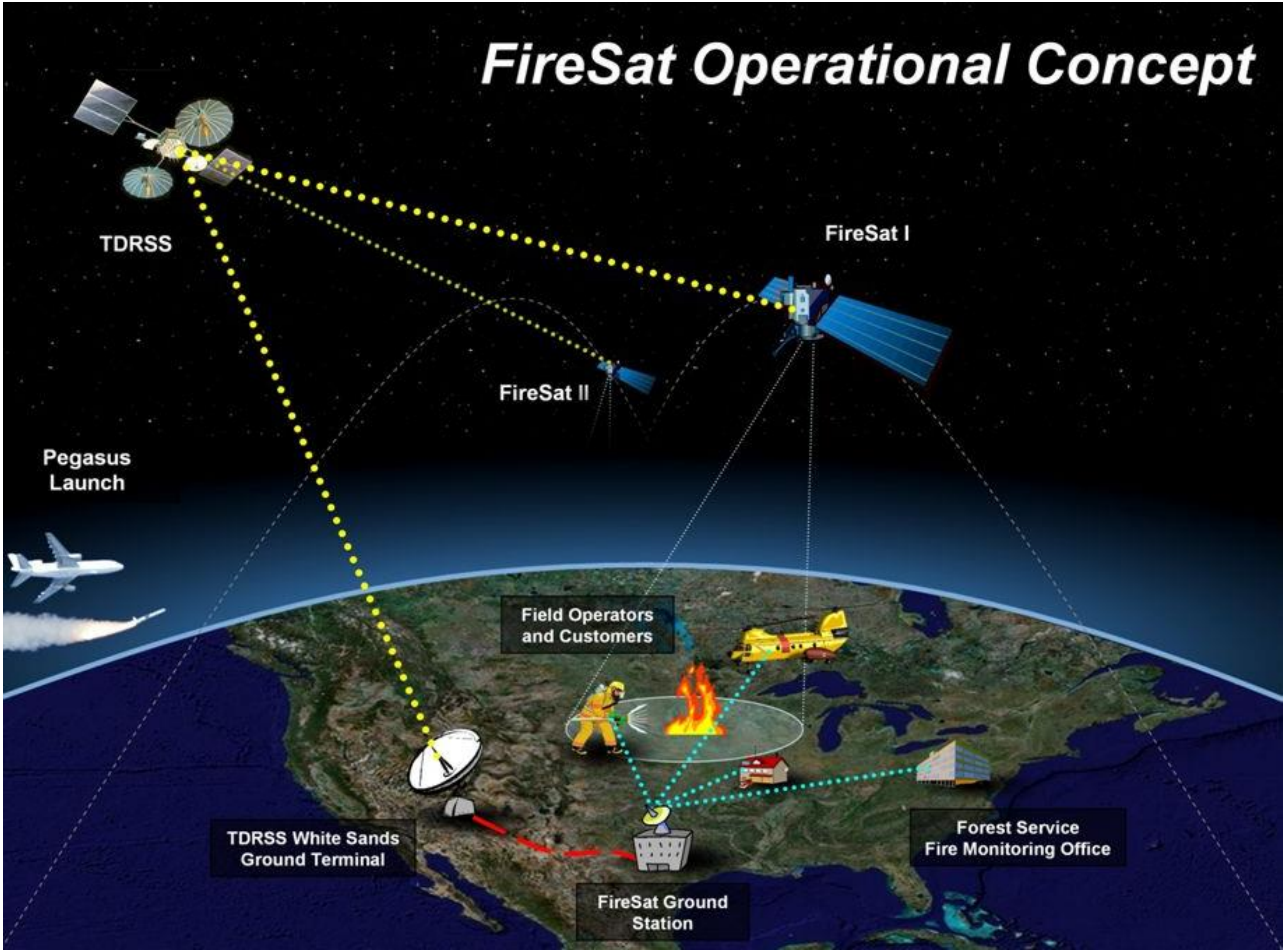
- **Operational goals** from the viewpoint of all stakeholders.
- **Overview of the System** of interest, including supporting systems.
- **Intended use** of the system during all life-cycle phases of the program/project, including but not limited to:
  - 1. Manufacturing and assembly / 2. Integration and test. / 3. Transportation and storage. / 4. Ground operations/launch integration. / 5. Launch Operations - launch, deployment, on-orbit checkout. / 6. Maintenance and disposal.
- Operational **timelines**.
- Command and data **architecture**.
- **End-to-end communication** strategy.
- **Integrated logistic** support (resupply, maintenance, assembly).
- Operational **facilities**.
- **Contingency and off-nominal** operations.

A ConOps does NOT include design solutions.

CONOPS Example: SPORT



Images: Charlie Gray


FireSat Operational Concept
TDRSS
FireSat I
FireSat II
Pegasus
Launch
Field Operators
and Customers
TDRSS White Sands
Ground Terminal
FireSat Ground
Station
Forest Service
Fire Monitoring Office

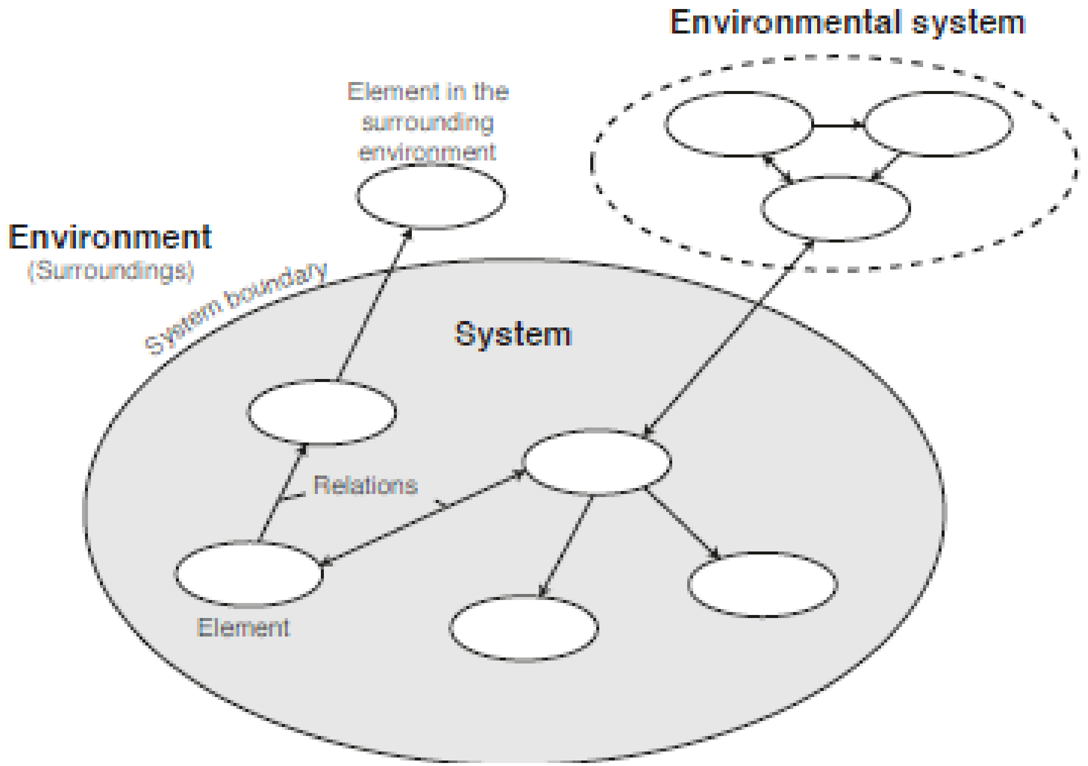
Functions

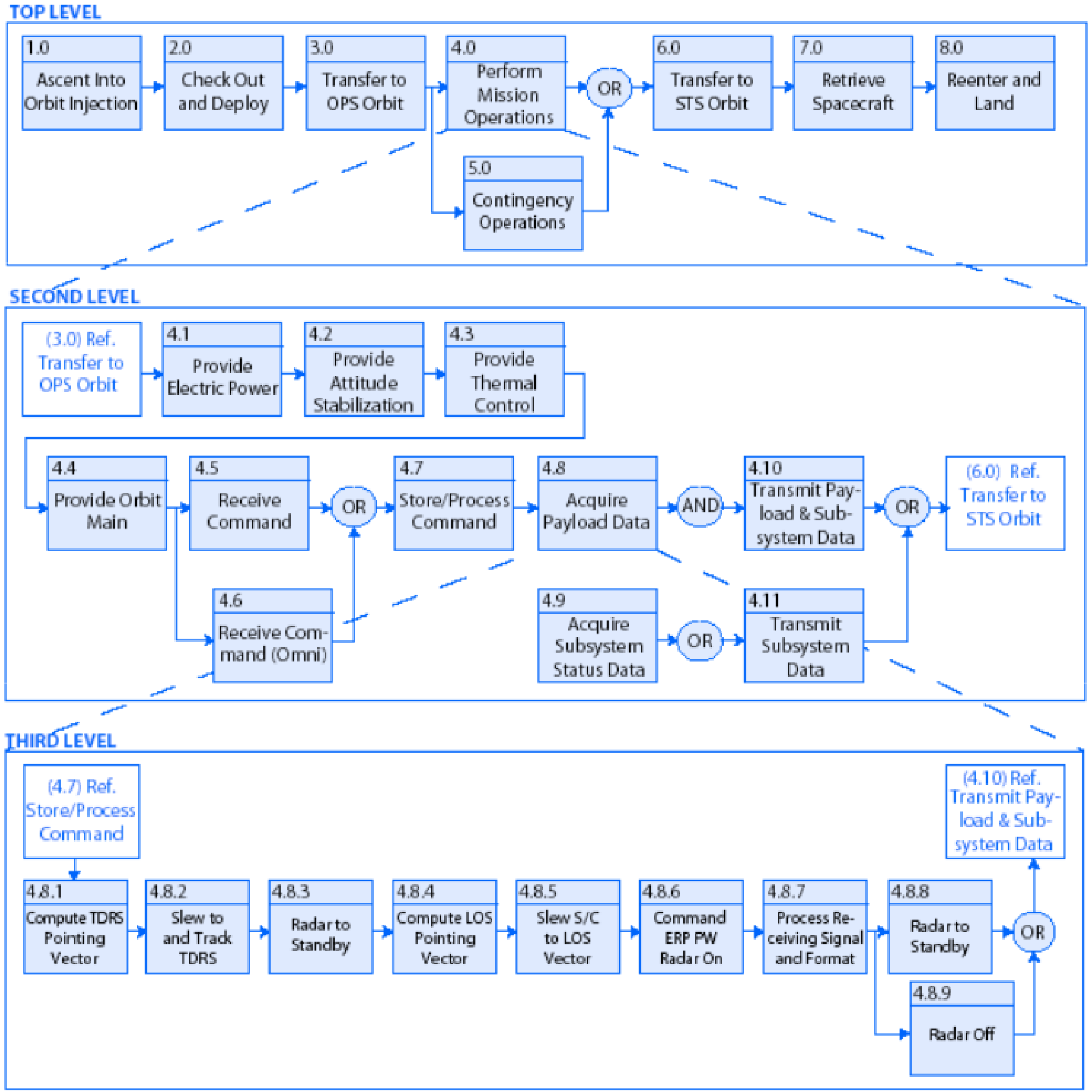
Function (action/activity)

- A function is an **action, an operation or a service**, performed by the system or one of its components, or also by an actor interacting with the system.
- Performing a **function generally produces exchange items** expected by other functions, and to do this, it requires other items provided by other functions.
- Several **functions can be grouped** into a mother function (they are then called subfunctions, or daughter functions, of this function). Symmetrically, a function can be refined into several functions.
- By convention, a function is named with a **verb**.

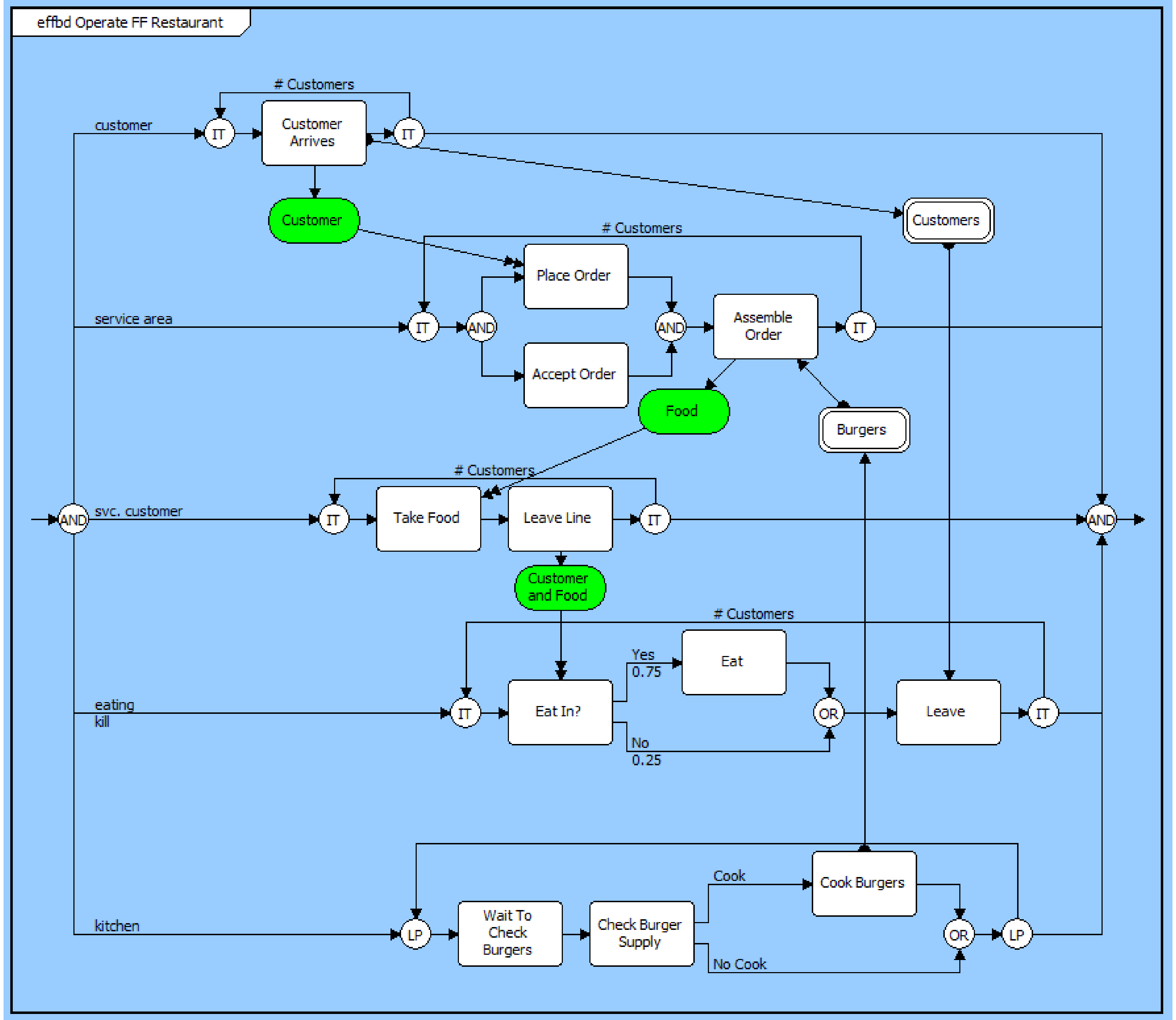
Boundaries

- Boundary is understood to be a more or less arbitrary **border between the system and its surroundings** or the environment in which it is embedded.





TOP LEVEL
1.0
Ascent Into Orbit Injection
2.0
Check Out and Deploy
3.0
Transfer to OPS Orbit
4.0
Perform Mission Operations
OR
6.0
Transfer to STS Orbit
7.0
Retrieve Spacecraft
8.0
Reenter and Land
5.0
Contingency Operations
SECOND LEVEL
(3.0) Ref. Transfer to OPS Orbit
4.1
Provide Electric Power
4.2
Provide Attitude Stabilization
4.3
Provide Thermal Control
4.4
Provide Orbit Main
4.5
Receive Command
OR
4.7
Store/Process Command
4.8
Acquire Payload Data
AND
4.10
Transmit Payload & Subsystem Data
OR
(6.0) Ref. Transfer to STS Orbit
4.6
Receive Command (Omni)
4.9
Acquire Subsystem Status Data
OR
4.11
Transmit Subsystem Data
THIRD LEVEL
(4.7) Ref. Store/Process Command
(4.10) Ref. Transmit Payload & Subsystem Data
4.8.1
Compute TDRS Pointing Vector
4.8.2
Slew to and Track TDRS
4.8.3
Radar to Standby
4.8.4
Compute LOS Pointing Vector
4.8.5
Slew S/C to LOS Vector
4.8.6
Command ERP PW Radar On
4.8.7
Process Receiving Signal and Format
4.8.8
Radar to Standby
OR
4.8.9
Radar Off



Enhanced Function Flow Block Diagram (EFFBD) (vitechcorp.com)

Requirements

requirement [ri-**kwahyuh** r-muh nt] SHOW IPA

SEE SYNONYMS FOR *requirement* ON THESAURUS.COM

noun

1 that which is required; a thing demanded or obligatory:
One of the requirements of the job is accuracy.

2 an act or instance of requiring.

3 a need or necessity:
to meet the requirements of daily life.

TO DO OR NOT TO DO

- **Functional Requirements** describe what the system should do and **Non-functional Requirements** place constraints on how these functional requirements are implemented.

IMPORTANCE OF HAVING GOOD REQUIREMENTS

- Requirements tell you **what the system needs to do** (functional requirements).
- **How well** the system needs to do it (performance requirements)
- **What environment** the system has to work in (environmental requirements).
- What the system **must do to fit into the bigger system** (interface requirements).
- What **lower level subsystems/assemblies/components must do to fit** into the system and make it all work (allocation of requirements/resources).
- What you need to **do before you fly** (verification activities).
- And basically, **when you are done** (requirements are met).

Architecture

- System architecture is the **embodiment of *concept***, the **allocation of physical/informational *function* to the elements of *form***, and the **definition of *relationships* among the elements** and with the **surrounding *context***.

One of the most important criteria for judging the goodness of a design:

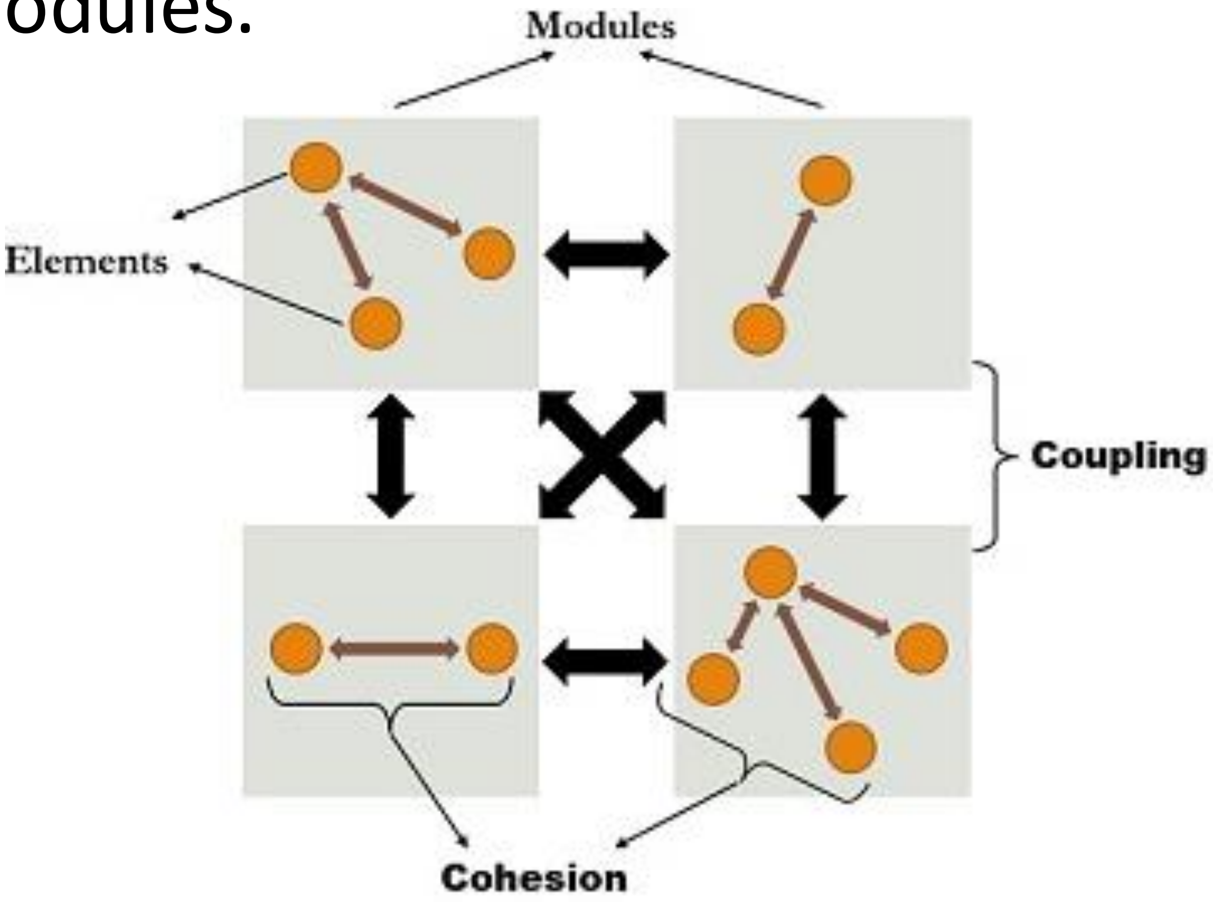
coupling and **cohesion**

together, these two concepts form the central theory of design.

Laplante, P. A. and Ovaska, S. J. Real-Time Systems Design and Analysis. 4th Ed. 2012.

Cohesion is about how well elements within a module belong together and serve a common purpose.

Coupling is about how much one module depends or interacts with other modules.

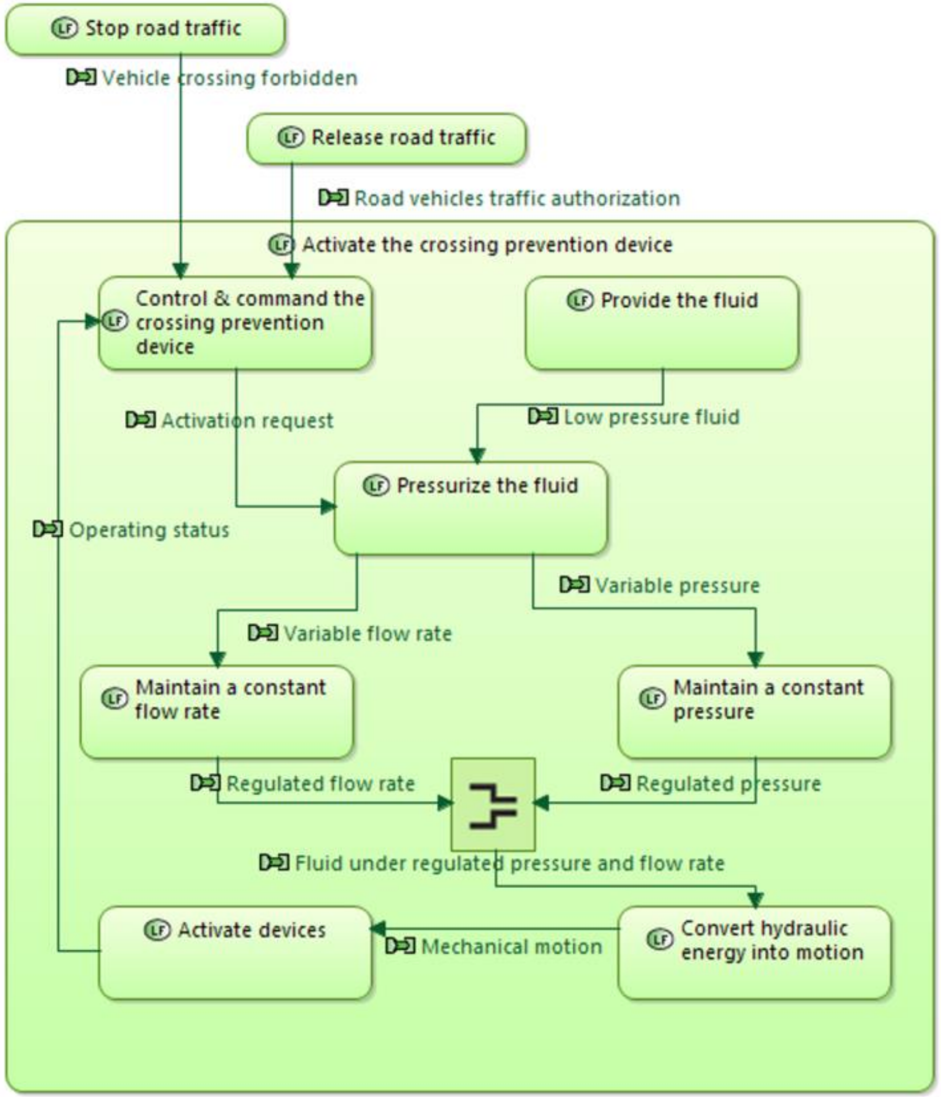


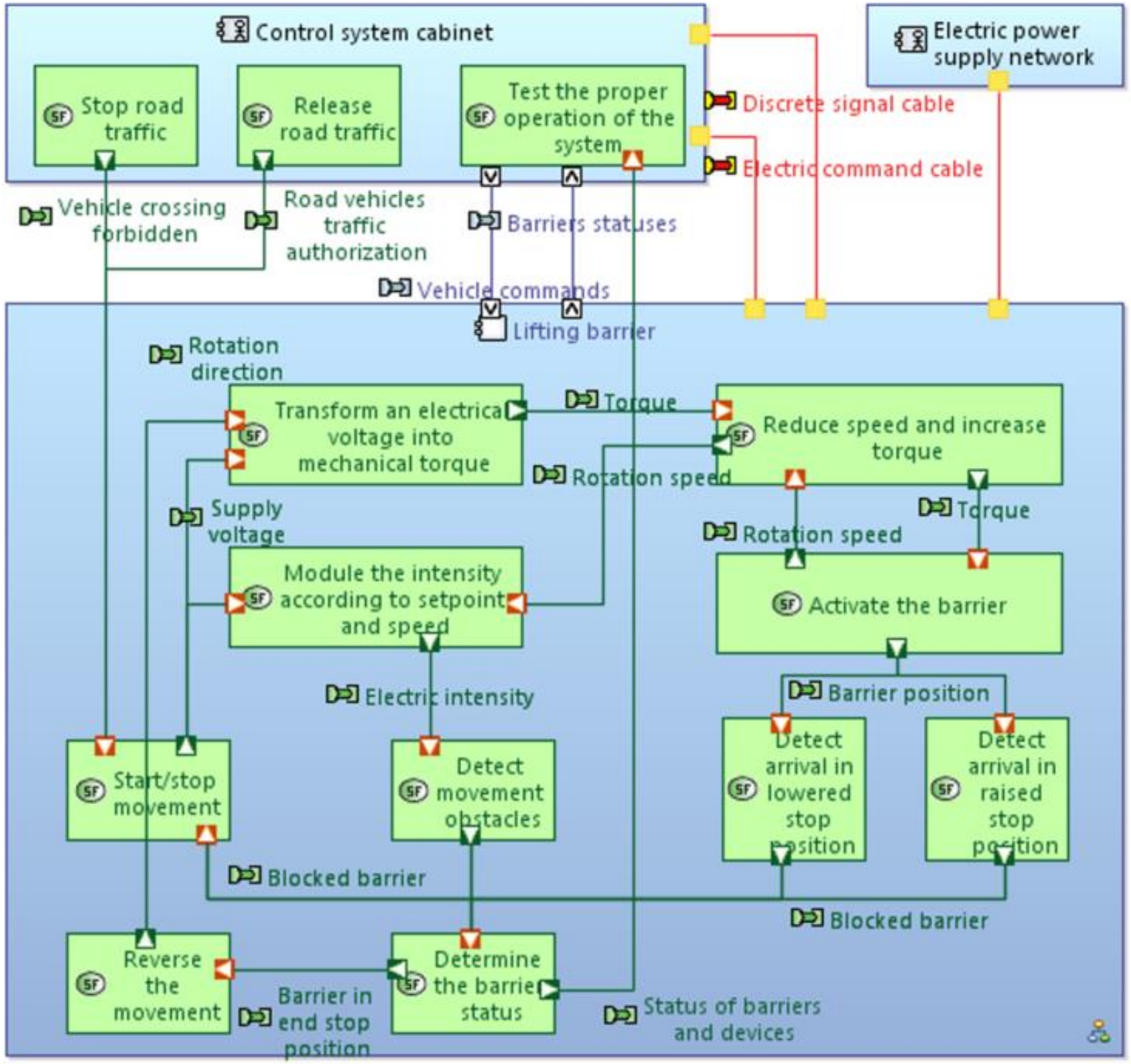
- When a **large system** is **decomposed** into **smaller entities**, it's inevitable that these entities will **interact with one another**.

- If the **boundaries** of these **entities** have been **poorly identified**, then the entities will **heavily depend** and **frequently interact** with **one another**.

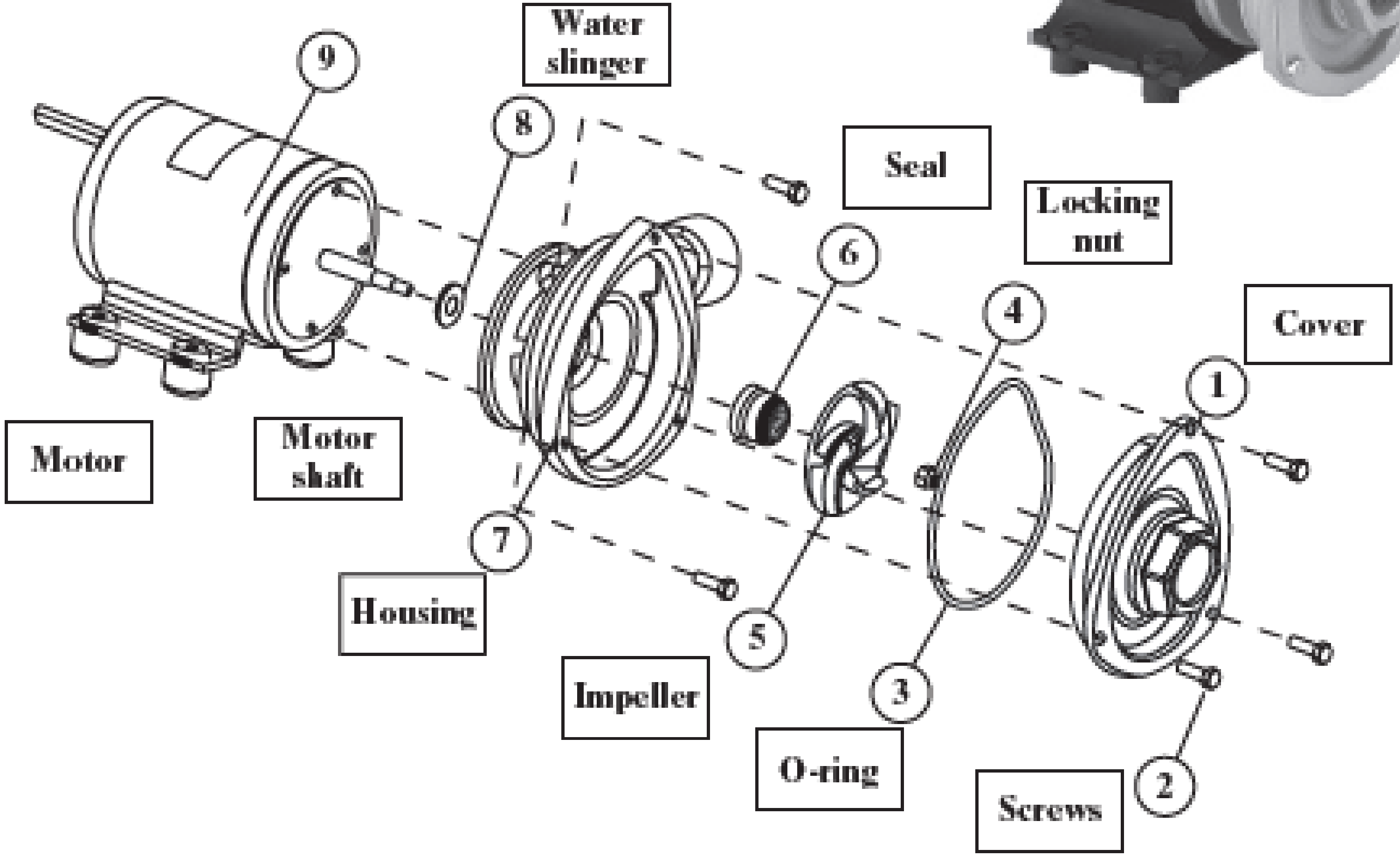
- In a **poor design**, it might also happen that **properties and functions within an entity** perform **diverse tasks** and therefore **don't seem to belong together**.

Examples


Stop road traffic
Vehicle crossing forbidden
Release road traffic
Road vehicles traffic authorization
Activate the crossing prevention device
Control & command the crossing prevention device
Provide the fluid
Activation request
Low pressure fluid
Pressurize the fluid
Operating status
Variable pressure
Variable flow rate
Maintain a constant flow rate
Maintain a constant pressure
Regulated flow rate
Regulated pressure
Fluid under regulated pressure and flow rate
Activate devices
Mechanical motion
Convert hydraulic energy into motion

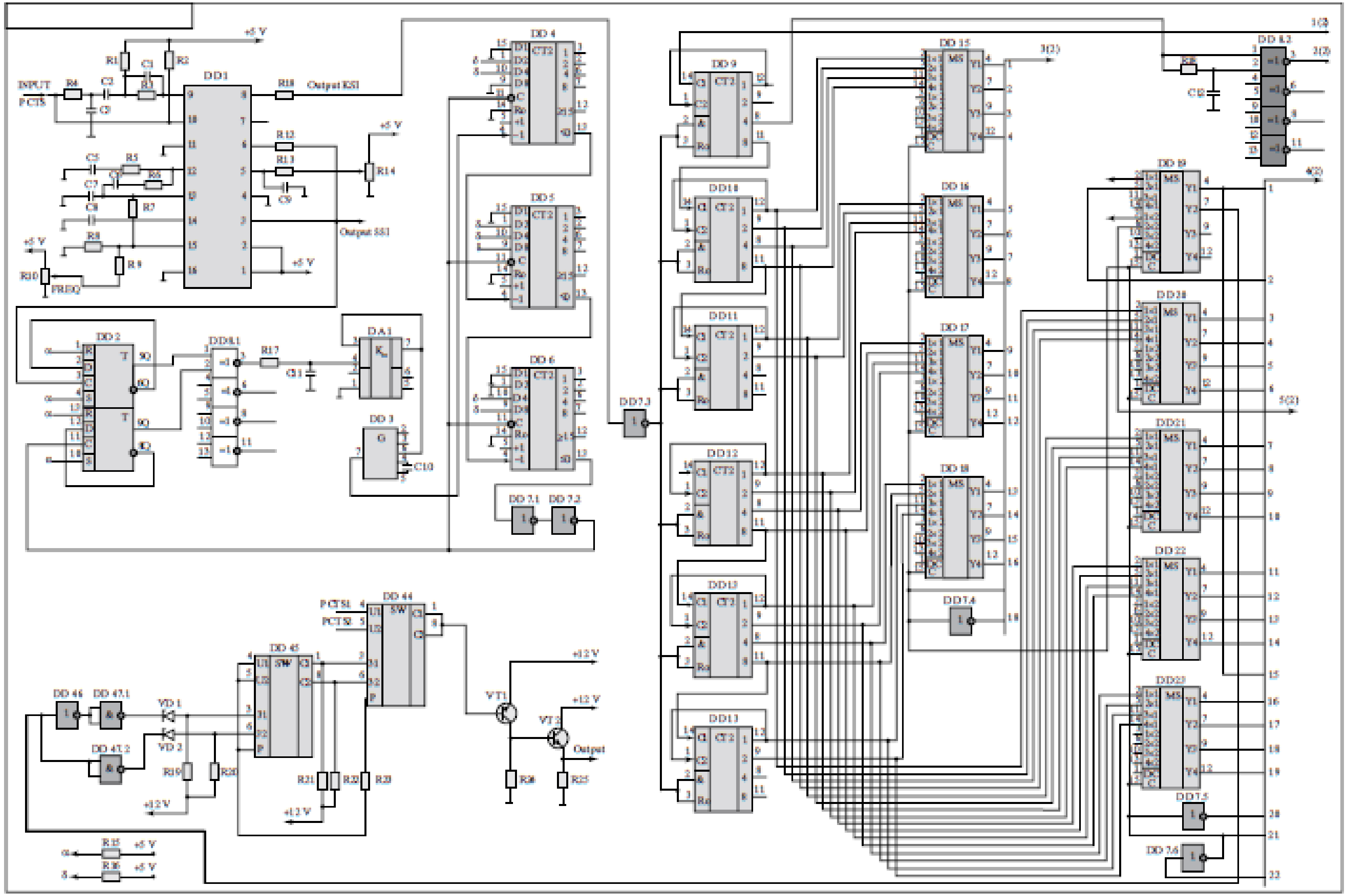

Control system cabinet
Electric power supply network
Stop road traffic
Release road traffic
Test the proper operation of the system
Discrete signal cable
Electric command cable
Vehicle crossing forbidden
Road vehicles traffic authorization
Barriers statuses
Vehicle commands
Lifting barrier
Rotation direction
Transform an electrical voltage into mechanical torque
Torque
Reduce speed and increase torque
Rotation speed
Supply voltage
Torque
Rotation speed
Module the intensity according to setpoint and speed
Activate the barrier
Electric intensity
Barrier position
Start/stop movement
Detect movement obstacles
Detect arrival in lowered stop position
Detect arrival in raised stop position
Blocked barrier
Blocked barrier
Reverse the movement
Determine the barrier status
Barrier in end stop position
Status of barriers and devices

EXAMPLE OF FORM: centrifugal pump



SYSTEM FORM

Example of form: circuit components



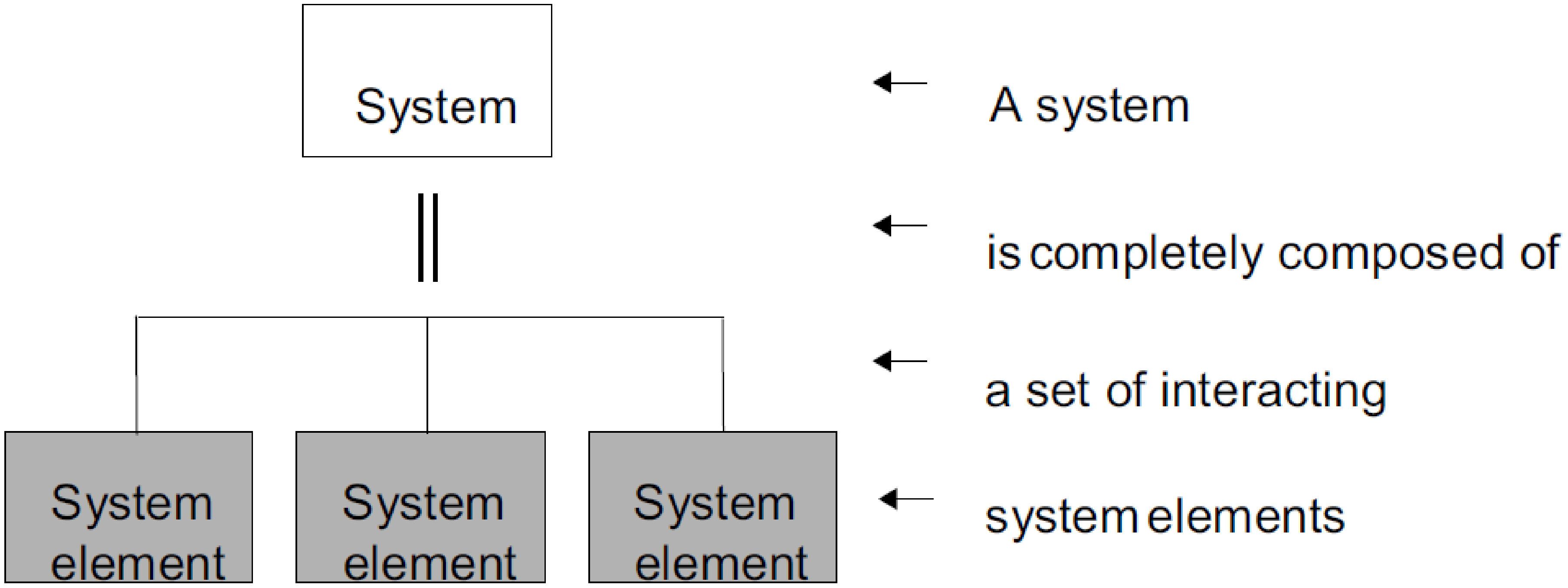
SYSTEM FORM

Example of form: software code

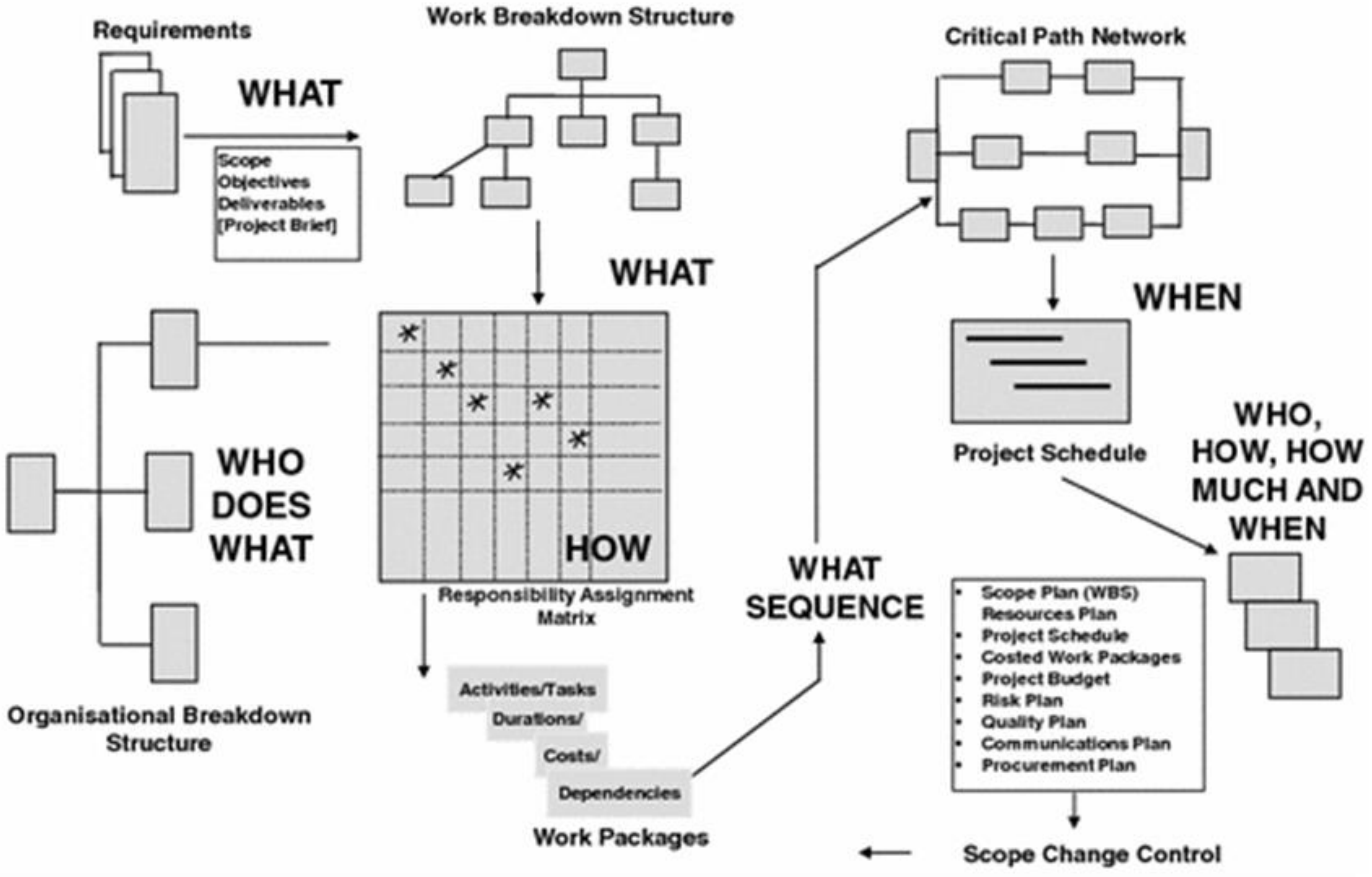
```
Procedure bubblesort (List array, number length_of_array)
    for i=1 to length_of_array - 1;
        for j=1 to length_of_array – 1;
            if array [j] > array [j+1] then
                temporary = array [j+1]
                array[j+1] = array [j]
                array[j] = temporary
            end if
        end of j loop
    end of i loop
return array
End of procedure
```

System Hierarchy

- System and system element relationship

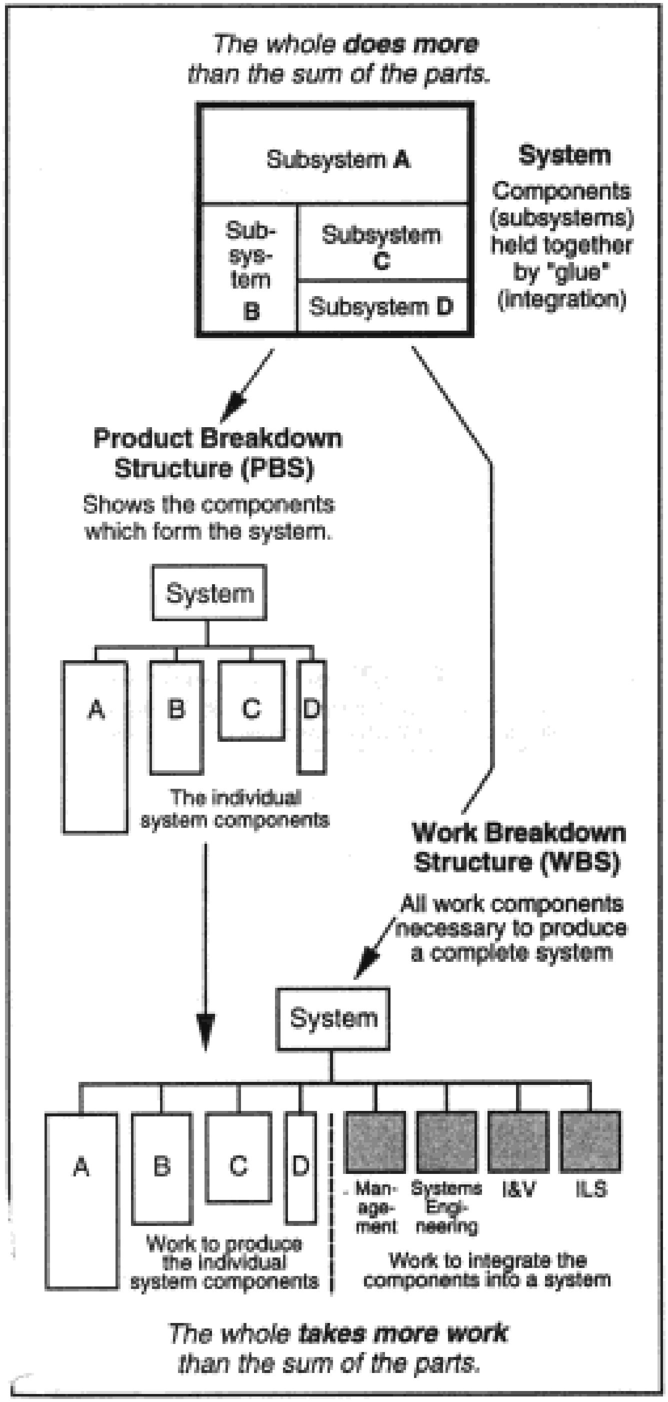


The system is decomposed into a hierarchy of smaller and smaller elements.



Requirements
WHAT
Scope
Objectives
Deliverables
[Project Brief]
Work Breakdown Structure
WHAT
Critical Path Network
WHEN
Project Schedule
WHO,
HOW, HOW
MUCH AND
WHEN
WHO
DOES
WHAT
HOW
Responsibility Assignment
Matrix
Organisational Breakdown
Structure
Activities/Tasks
Durations/
Costs/
Dependencies
Work Packages
WHAT
SEQUENCE
Scope Plan (WBS)
Resources Plan
Project Schedule
Costed Work Packages
Project Budget
Risk Plan
Quality Plan
Communications Plan
Procurement Plan
Scope Change Control

- A Work Breakdown Structure (WBS) is a **hierarchical breakdown of the work** necessary to complete a project. The WBS should be a product-based, hierarchical division of deliverable items and associated services. As such, **it should contain the project's Product Breakdown Structure (PBS),** with the specified prime product(s) at the top, and the systems, segments, subsystems, etc. at successive lower levels.
- The **WBS is built from the PBS** by adding, at each branch point of the PBS, any necessary service elements such as **management, systems engineering, integration and verification (I&V), and integrated logistics support (ILS).**



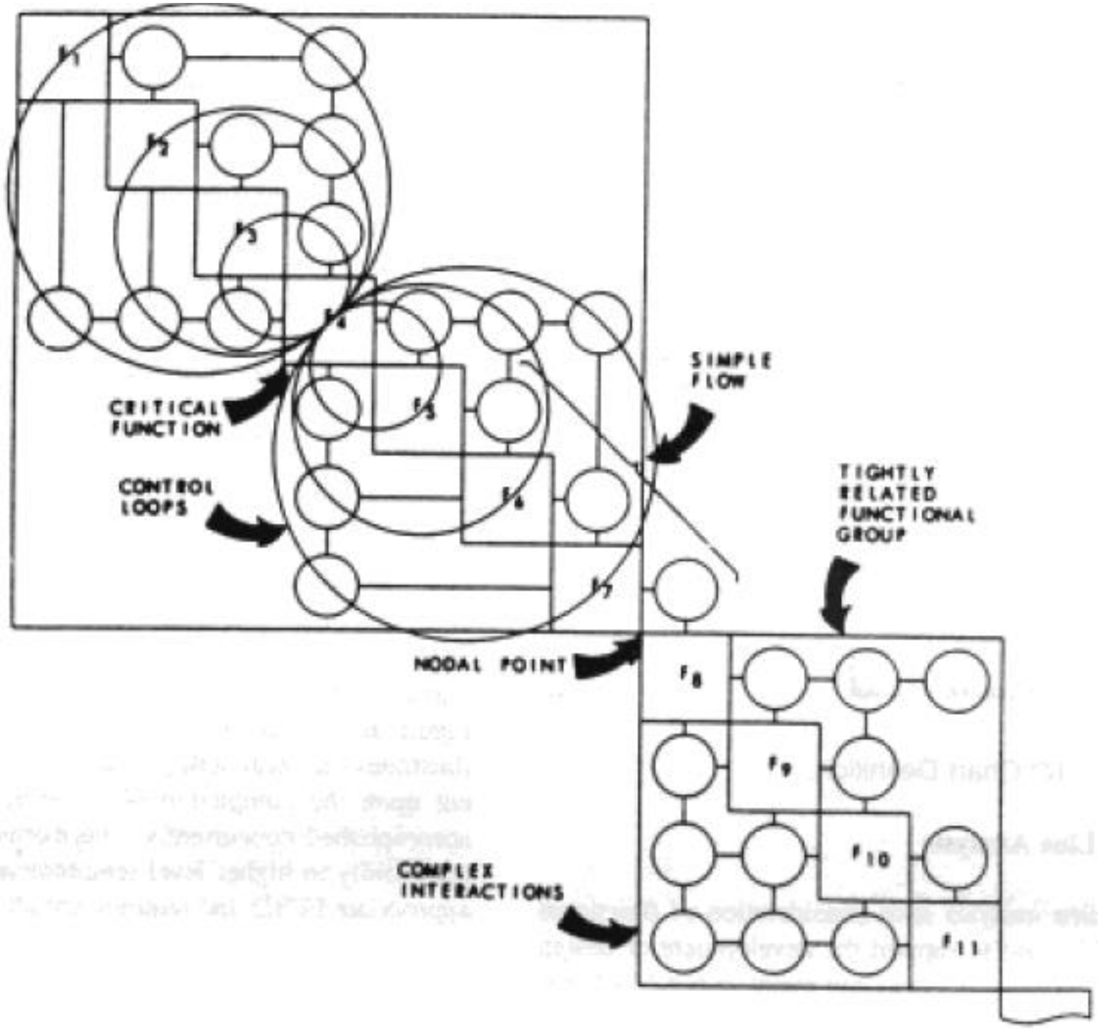
It is important to identify the interfaces.

- Complex systems have many interfaces
  - Common interfaces reduce complexity
  - System architecture drives the types of interfaces to be utilized in the design process
  - Clear interface identification and definition reduces risk
  - Most of the problems in systems are at the interfaces.
  - Verification of all interfaces is critical for ensuring compatibility and operation

N2 Analysis

- N2 Analysis is a tool that uses a nxn matrix to record the interconnections between elements of a system. It has a number of potential uses:
  - In system design to assess the degree of binding and coupling in a system and thereby determine candidate architectures based on the natural structure of the system.
  - In systems design to record, and thence aid the management of, the interfaces in a system.
  - In systems analysis to identify and document the interconnectivity in a system to help understand observed behavior and to provide guidance for improvement.

- Alternatively, the use of circles and numbers permits a separate listing of the data interfaces.
- The clockwise flow of data between functions that have a feedback loop can be illustrated by a larger circle called a control loop.
- The identification of a critical function, where function F4 has a number of inputs and outputs to all other functions in the upper module.
- A simple flow of interface data exists between the upper and lower modules at functions F7 and F8.
- The lower module has complex interaction among its functions.
- The N2 chart can be taken down into successively lower levels to the hardware and software component functional levels. In addition to defining the data that must be supplied across the interface, the N2 chart can pinpoint areas where conflicts could arise.



Final Considerations

Verification

- **VERIFICATION** testing relates back to the **approved requirements** set and can be performed **at different stages in the product life cycle**.
  - The approved specifications, drawings, parts lists, and other configuration documentation establish the configuration baseline of that product, which may have to be modified at a later time.
  - Without a verified baseline and appropriate configuration controls, later modifications could be costly or cause major performance problems.

Validation

- **VALIDATION** relates back to the **CONOPs document**.
- **VALIDATION** testing is conducted **under realistic conditions (or simulated conditions)** on **end products** for the purpose of determining the effectiveness and suitability of the product for **use in mission operations by typical users**.
- **VALIDATION** can be performed in each development phase using phase products (e.g., models) and not only at delivery using end products.