CLASSICAL DIAGRAMS

WEEK	CLASS ACTIVITY	REF	INDIVIDUAL	W	GROUP	W
1 28/Jul	Course Structure and Initial Definitions Systems Engineering Review	[illegible]	IA-01 - Reading and Conceptual Questions (10)	10%		0%
2 04/Aug	Classical Systems Engineering Diagrams (IDEF-0/N2/eFFBD/DFD)	[4] *papers	IA-02	0%	GA-02 - Prepare a representation of your system using classical Diagrams	50%
3 11/Aug	Transition from Legacy to MBSE MBSE Methodologies MBSE Languages	[5] [7] *papers	IA-03 - Reading and Conceptual Questions (10)	10%		0%
4 18/Aug	OPM - Basic	[6]	IA-04 - Exercises	10%		0%
5 25/Aug	OPM - Extended	[6]	IA-05 - Exercises	10%		0%
6 01/Sep	OPM - Group Presentation		IA-06	0%	G6 - Prepare a presentation of your system using OPM	50%
7 08/Sep	SysML Introduction (bdd/ibd)	[7]	IA-07 - Exercises	10%		0%
8 15/Sep	P1 - Conceptual Questions and Case	[1] [2] [3] [4] [6]	IA-08 - Questions and a mini-case	50%	GA-08 -	
				100%		**100%**

enhanced Functional Flow Block Diagram

FUNCTION DEFINITION

- **Function** is the activity, operation, or transformation that **causes or contributes to performance**.
- In designed systems, function is the **actions for which a system exists**, which ultimately lead to the delivery of value.
- **Function is executed by form**, which is instrumental in function.
- **Function emerges from functional interaction** between entities. Function is a product/system attribute.

- *Function is about activity, in contrast with form, which is about existence.*

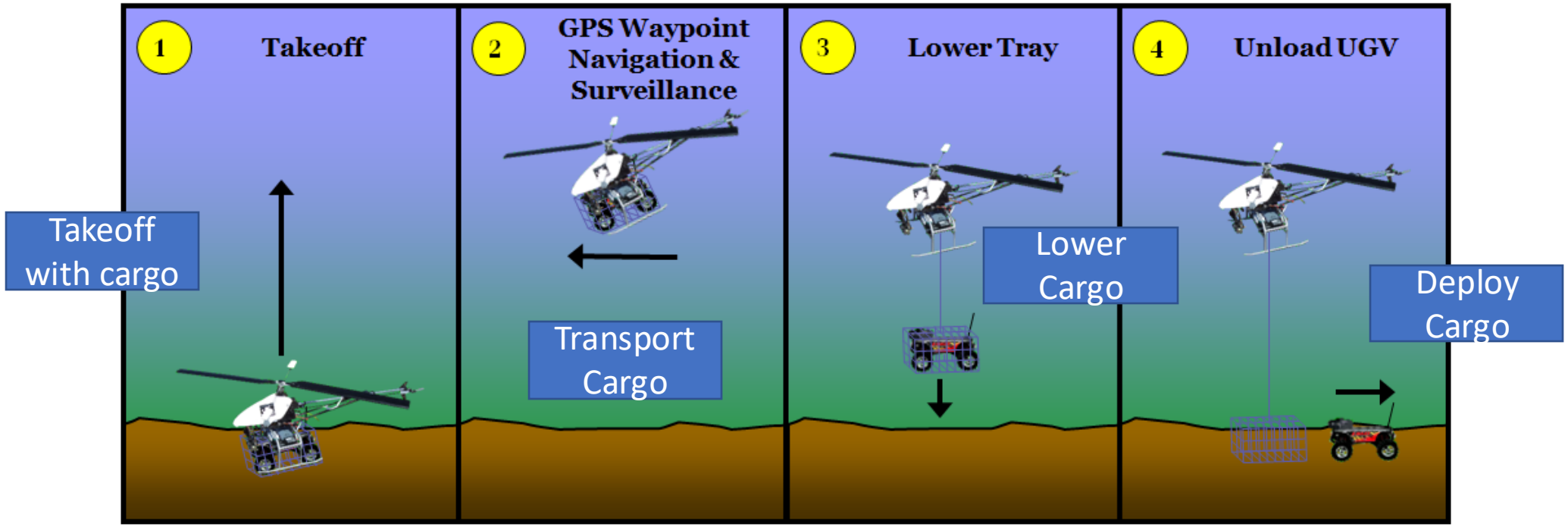
Functional modelling

- Functional modelling in Systems Engineering is a **structured representation of functions** (i.e. activities, actions, processes, operations) within the modelled system.
- The purpose of the function model is to **describe the functions and processes**, **assist with discovery of information needs**, help **identify opportunities** and **establish a basis for determining product and service costs**. In Systems Engineering, a function model is created from a functional modelling perspective.

Functional architecture

- The functional architecture is (usually) a **top-down decomposition** of system functional and performance requirements.
- The architecture will show not only the functions that have to be performed, but also the **logical sequencing of the functions and performance requirements associated with the functions**.
- The **functional architecture** produced by the Functional Analysis and Allocation process is the detailed package of documentation developed to analyze the functions and allocate performance requirements.

From CONOPs (1/2)



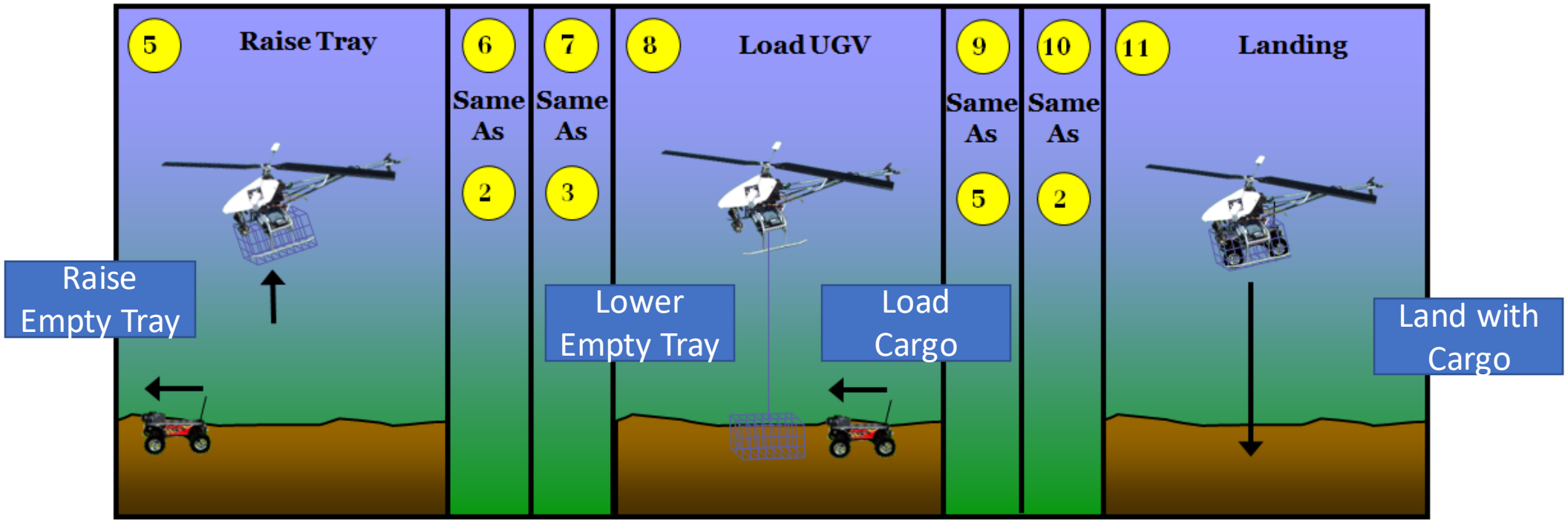
Autonomous takeoff is integral to Rotomotion's control package, but requires proper calibration of sensor gains.

Autonomous hovering and GPS waypoint navigation is integral to Rotomotion's control package, but requires a collision avoidance system for true safety of flight.

The cargo must be lowered at sufficient speed to avoid possible buildup of oscillations.

Stable hovering is especially important while unloading the UGV.

From CONOPs (2/2)



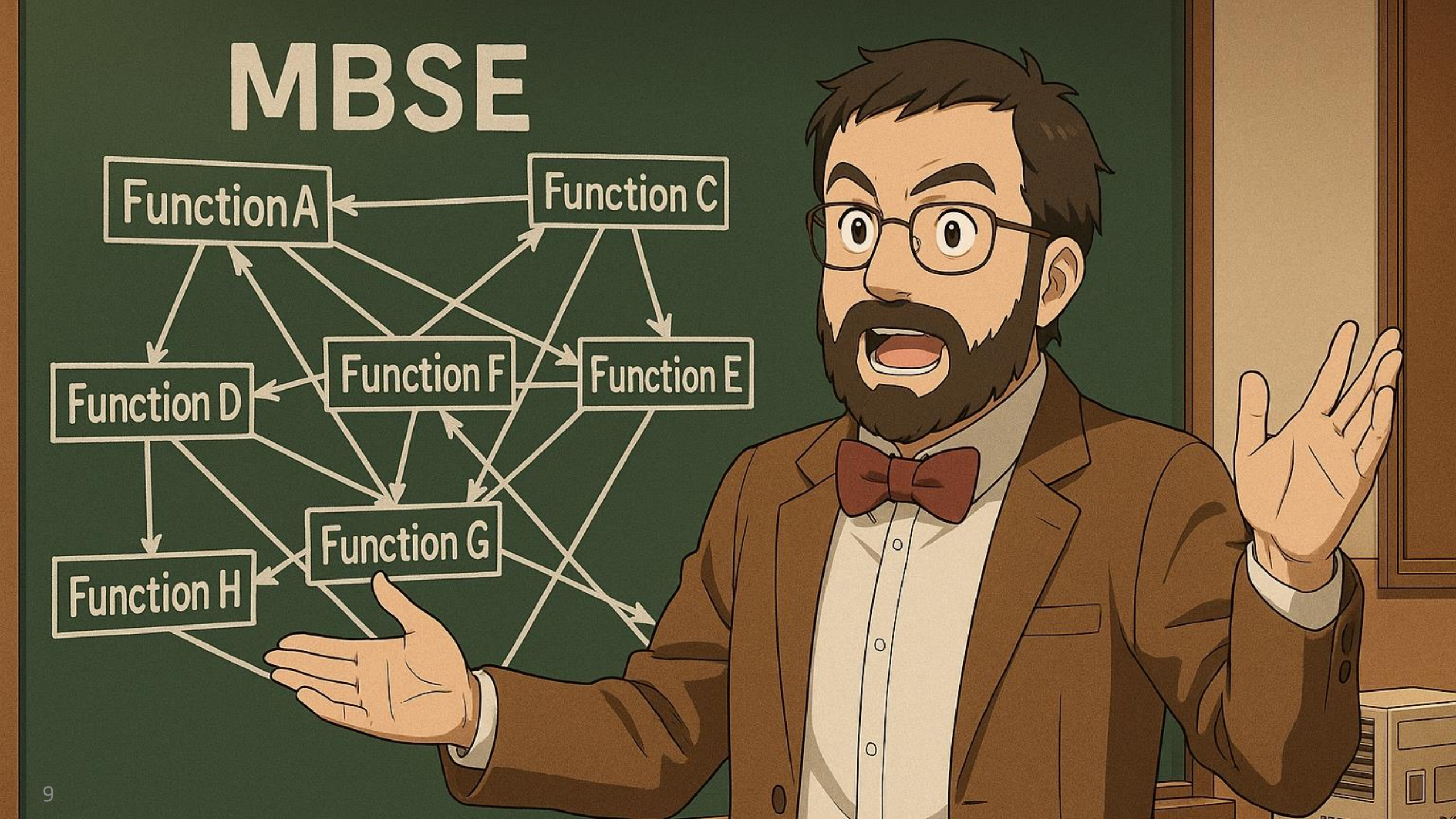
The cargo must be raised at sufficient speed to avoid possible buildup of oscillations, but less weight during raising means less risk.

(6) GPS Waypoint Navigation & Surveillance
(7) Lower Tray

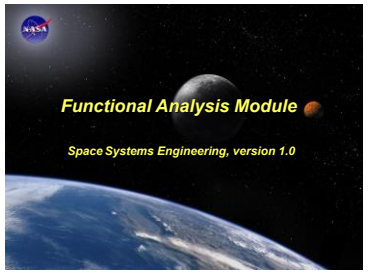
Stable hovering is especially important while loading the UGV. Servoing of UGV into the tray requires accurate sensors and failsafe tray design.

(9) Raise Tray
(10) GPS Waypoint Navigation & Surveillance

Autonomous landing is integral to Rotomotion's control package, but requires identification of proper landing zone through sensor systems for truly reliable landings.

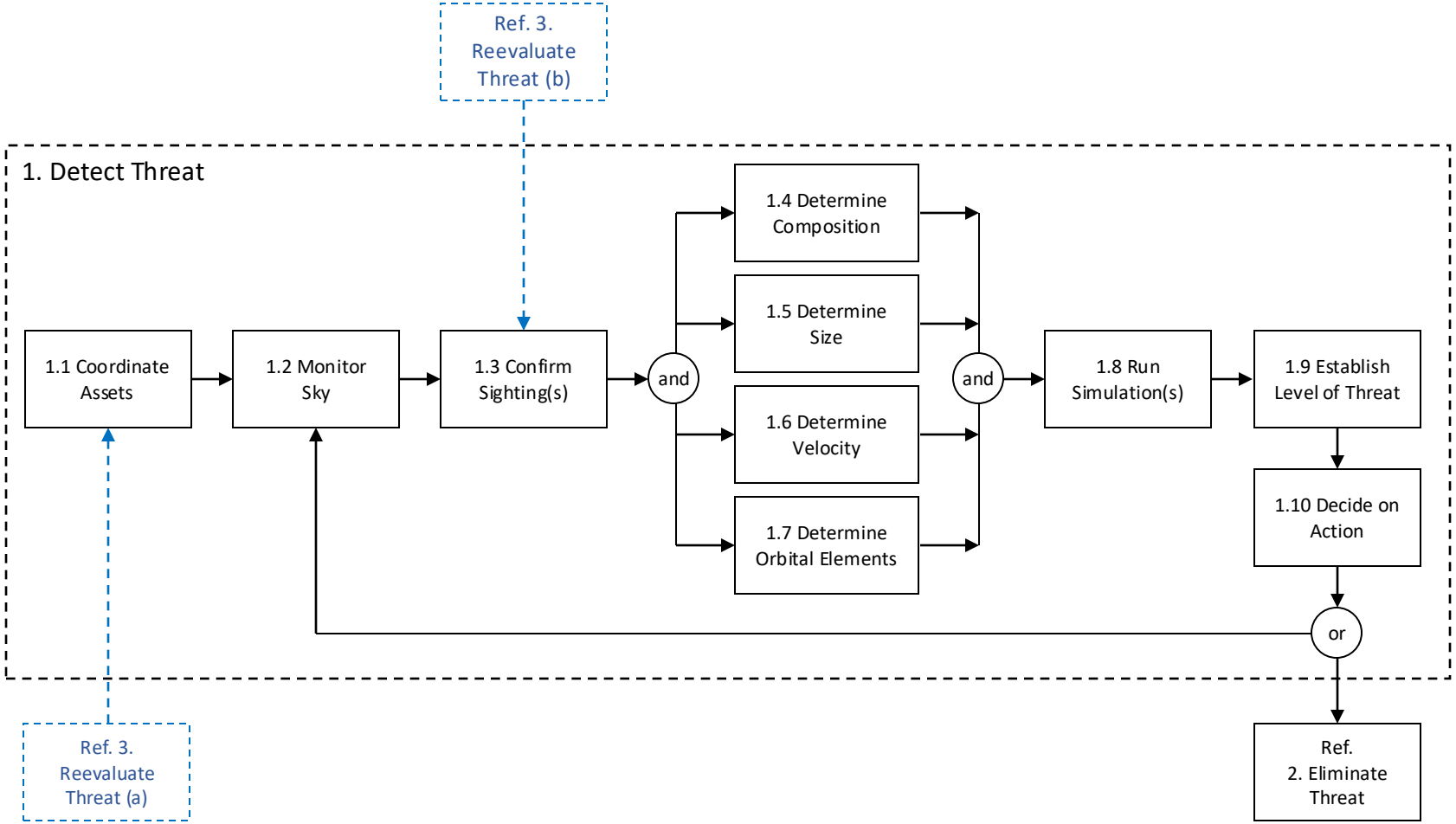

MBSE
Function A
Function C
Function D
Function F
Function E
Function G
Function H

Planetary Defense Program

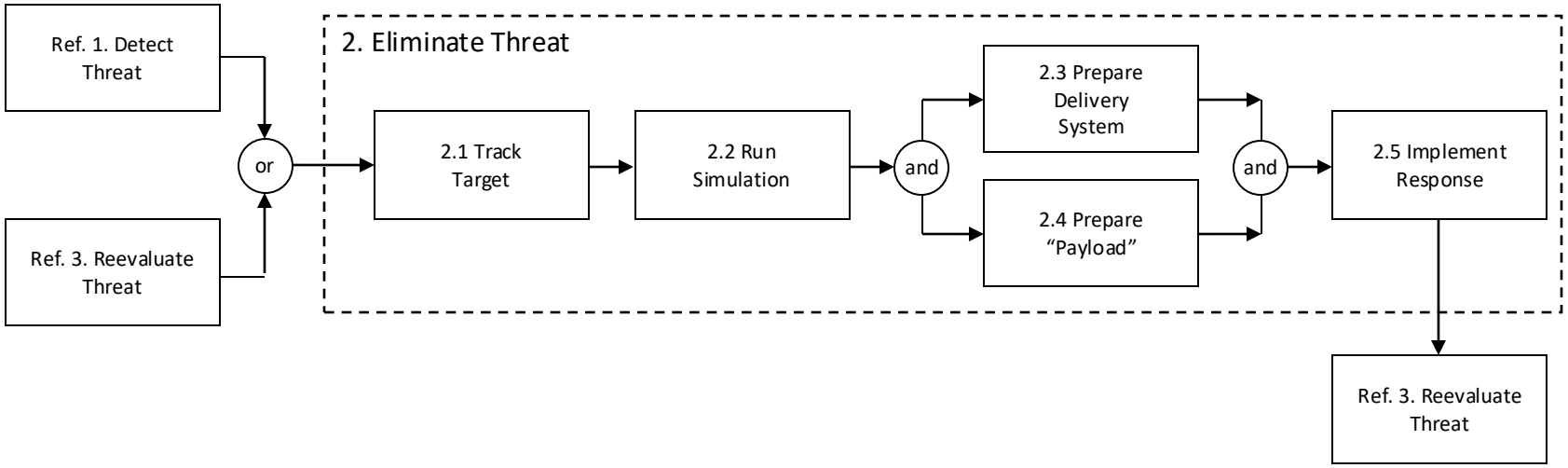


- Planetary Defense
  - 1. Detect Threat
    - 1.1 Coordinate Assets
    - 1.2 Monitor Sky
    - 1.3 Confirm Sighting(s)
    - 1.4 Determine Composition
    - 1.5 Determine Size
    - 1.6 Determine Velocity
    - 1.7 Determine Orbital Elements
    - 1.8 Run Simulation(s)
    - 1.9 Establish Level of Threat
    - 1.10 Decide on Action
  - 2. Eliminate Threat
    - 2.1 Track Target
    - 2.2 Run Simulation(s)
    - 2.3 Prepare Delivery System
    - 2.4 Prepare “Payload”
    - 2.5 Implement Response
  - 3. Re-evaluate Threat
    - 3.1 Monitor Impact
    - 3.2 Ref. Detect Threat
    - 3.3 Confirm Success/Failure

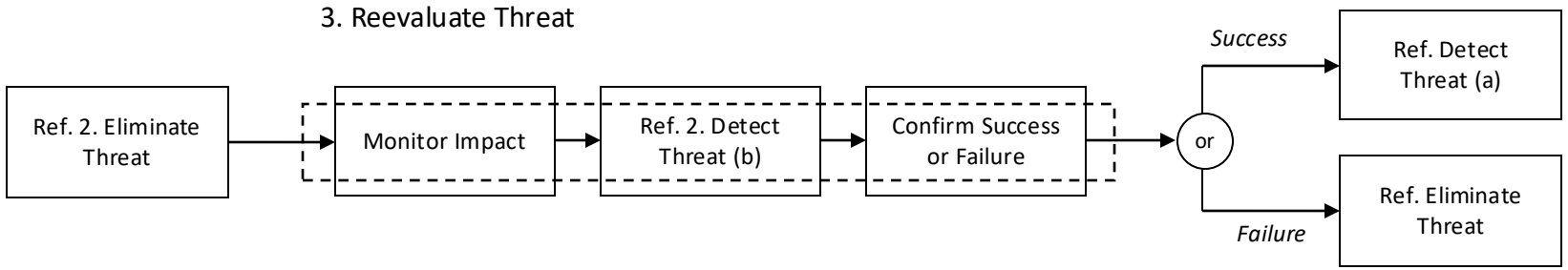
Planetary Defense Level 1 Functional Flow Block Diagram For Threat Detection



Planetary Defense Level 1 Functional Flow Block Diagram For Threat Elimination



Planetary Defense Level 1 Functional Flow Block Diagram For Threat Reevaluation



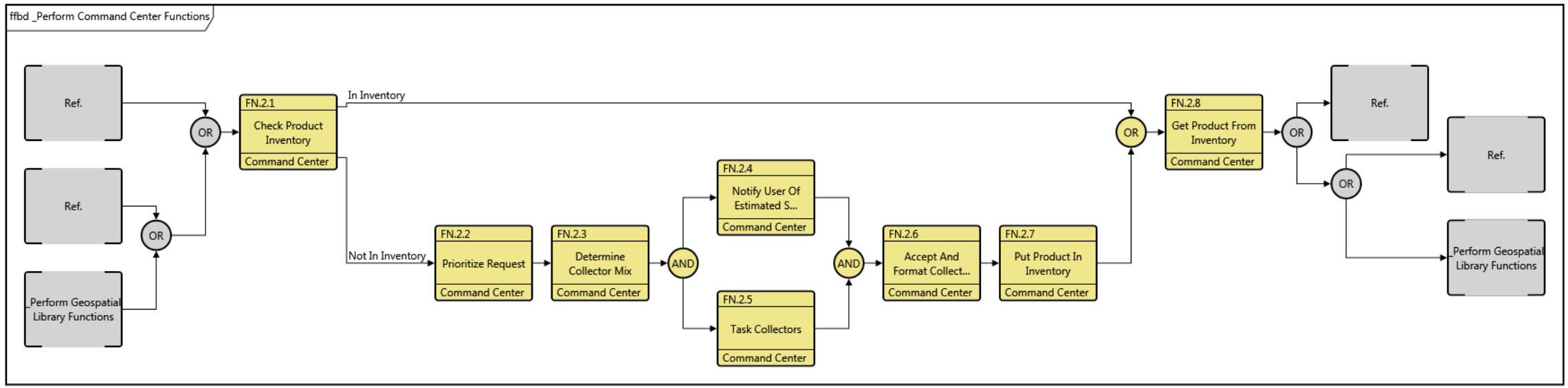
The FFBD – Functional Flow Block Diagram

- The Functional Flow Block Diagram (FFBD) was developed in the 1950s by TRW Inc., a U.S. defense contractor.
- **TRW created the FFBD** as a structured method to **represent the sequential flow of system functions**, particularly for complex aerospace and defense projects.
- In the 1960s, NASA adopted FFBDs to visualize the time sequence of events in space systems and flight missions.

FFBD – Functional Flow Block Diagram

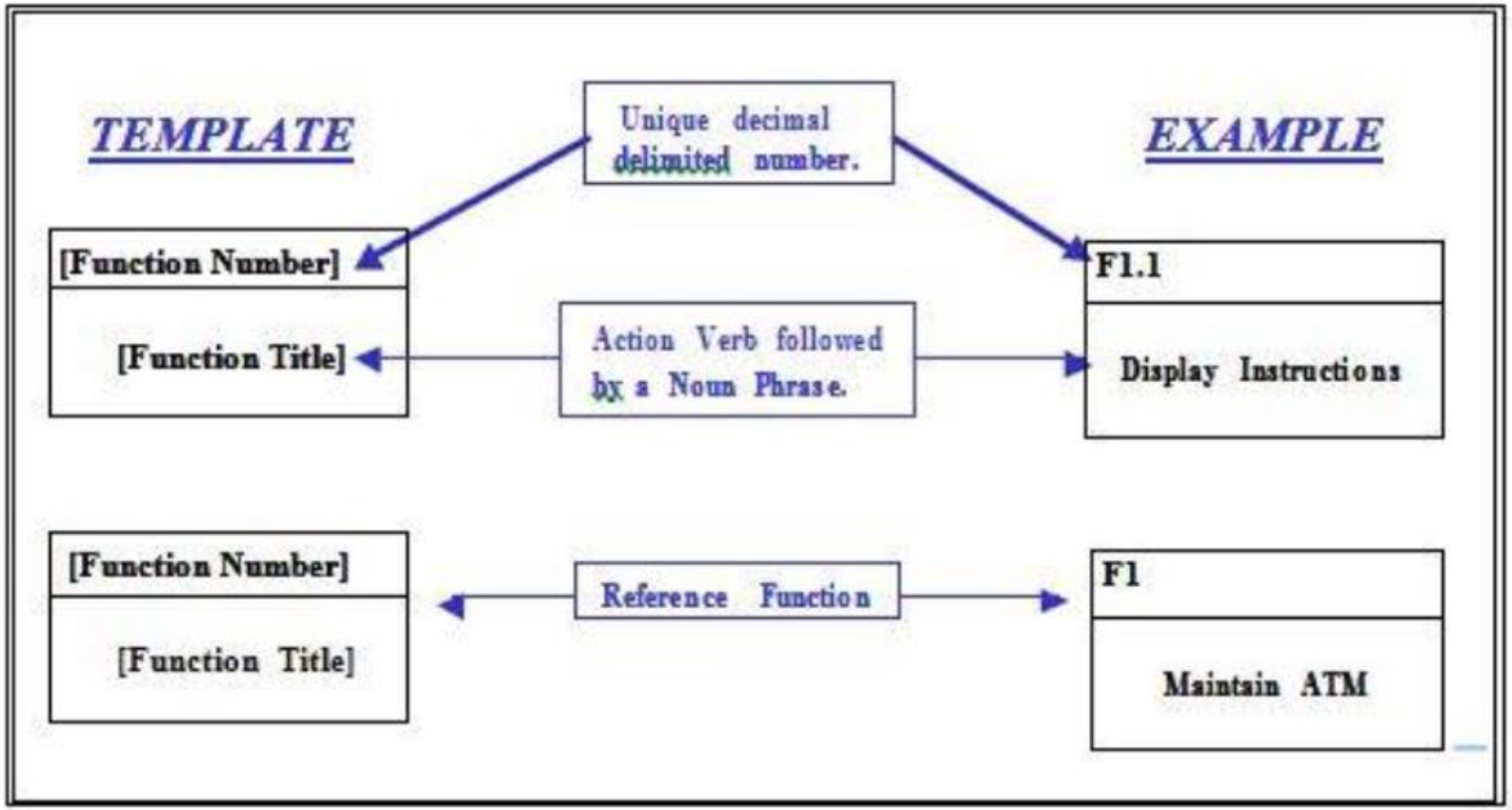
- The FFBD is a **multi-layered, sequenced diagram** of the functional flow of a system.
- An FFBD usually defines the sequences and **supports step-by-step detailing** of systems, but it can also be used effectively to define processes when developing and producing systems.
- In the FFBD method, they are organized and represented by their **logical order of execution**.
- A key concept in modeling is

that for a function to begin, the previous function or functions within the "control" flow must have ended. For example, a "view targets" function would logically not start until a "detect targets" function was completed.


ffbd _Perform Command Center Functions
Ref.
Ref.
_Perform Geospatial Library Functions
OR
OR
FN.2.1
Check Product Inventory
Command Center
In Inventory
Not In Inventory
FN.2.2
Prioritize Request
Command Center
FN.2.3
Determine Collector Mix
Command Center
AND
FN.2.4
Notify User Of Estimated S...
Command Center
FN.2.5
Task Collectors
Command Center
AND
FN.2.6
Accept And Format Collect...
Command Center
FN.2.7
Put Product In Inventory
Command Center
OR
FN.2.8
Get Product From Inventory
Command Center
OR
Ref.
OR
Ref.
_Perform Geospatial Library Functions

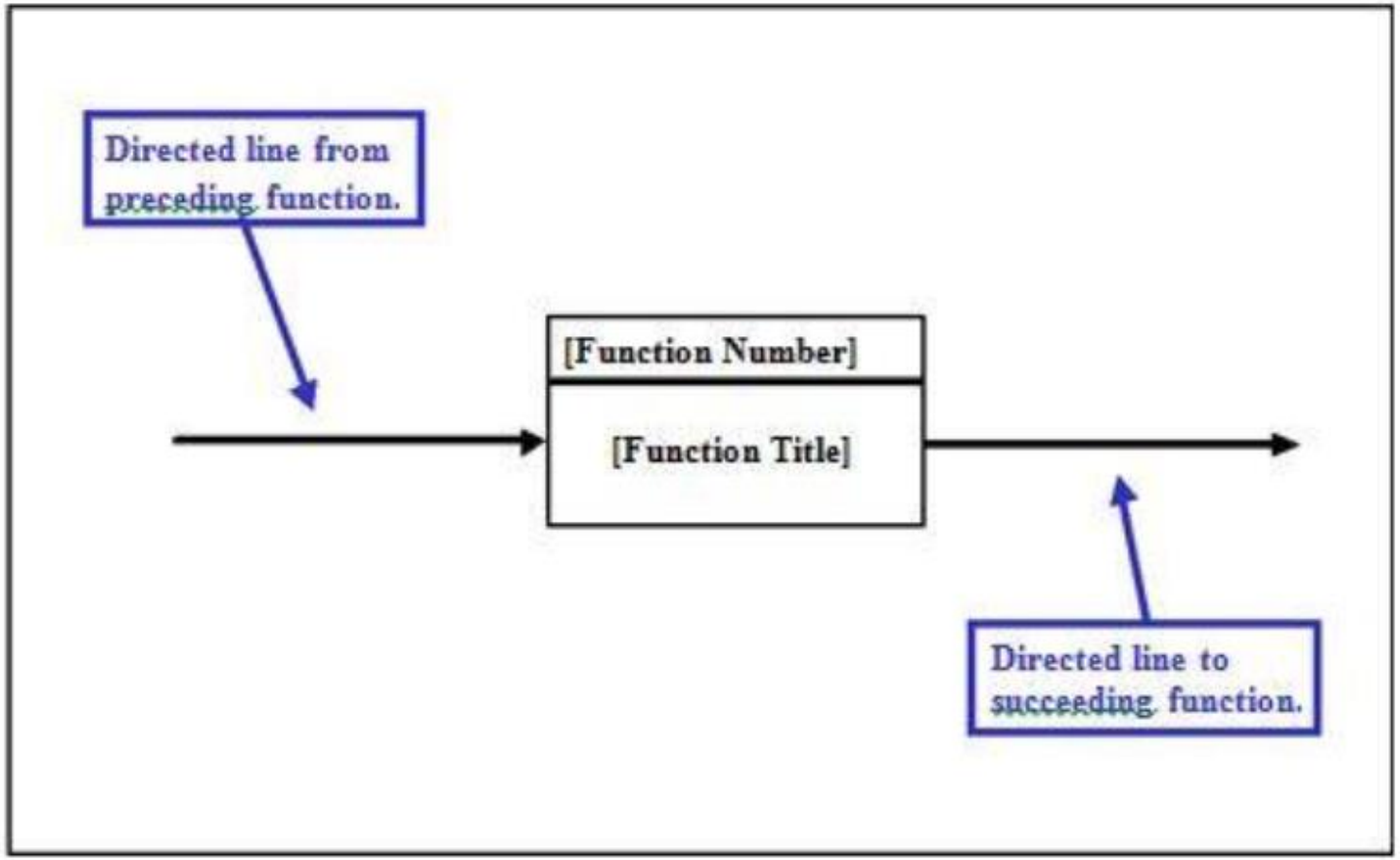
Symbology

- A function must be represented by a rectangle containing the function title (a **verb** followed by a noun) and its unique number.



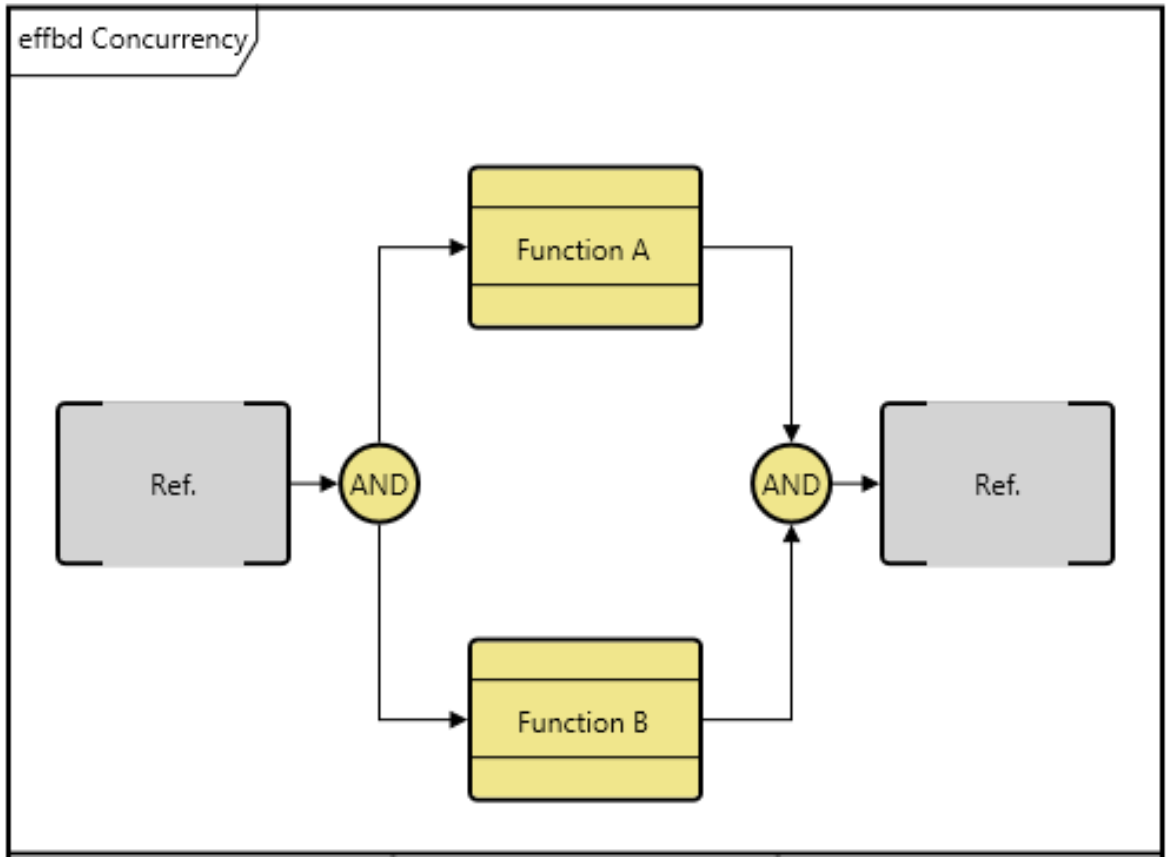
Functional flow

- A line with a single arrowhead should represent the functional flow from left to right.



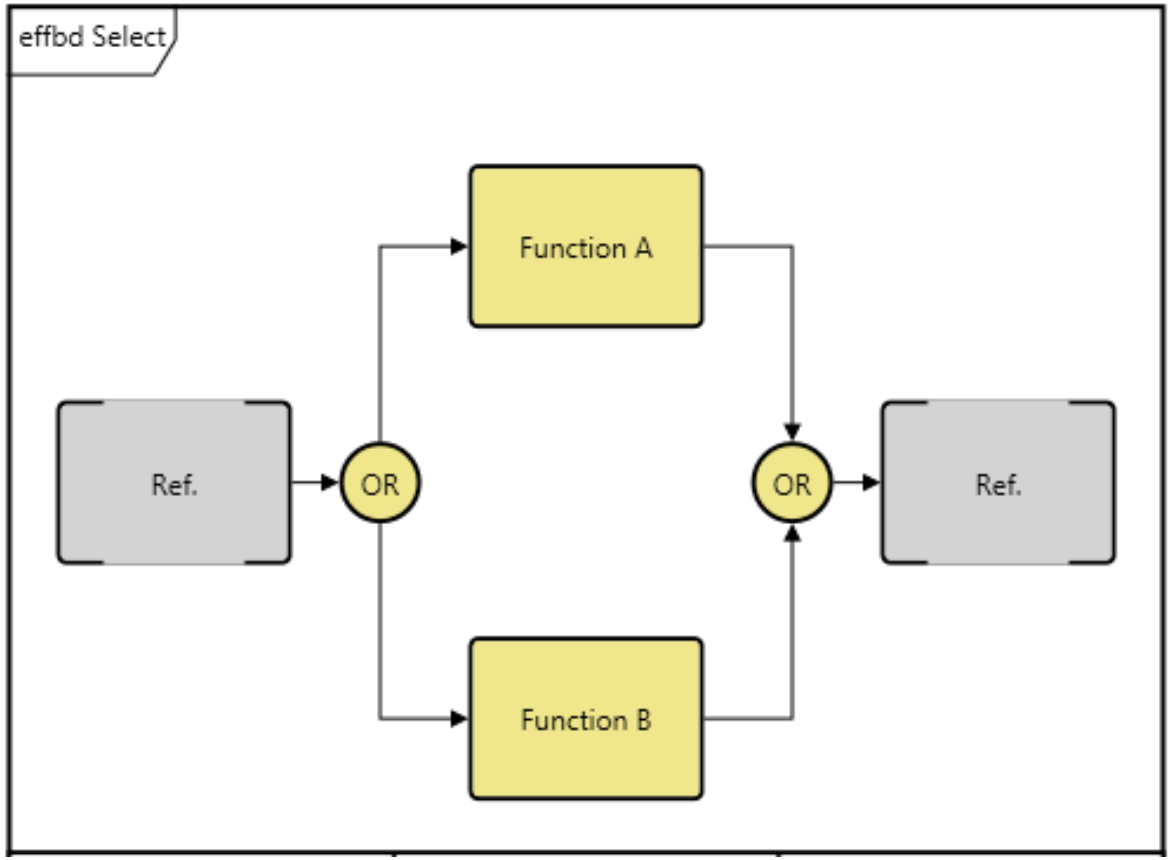
Logical conditions: and

- A parallel construct consists of an AND node, followed by separate branches that rejoin and terminate at another matching AND node.
- Each branch can contain any number of functions and control constructs.
- When executed, the first entity on each branch will be enabled at the same simulation clock time. The construct cannot be exited (from the second AND node) until all branches have completed their processing.
- Control is then passed to the next function or construct after the parallel construct.



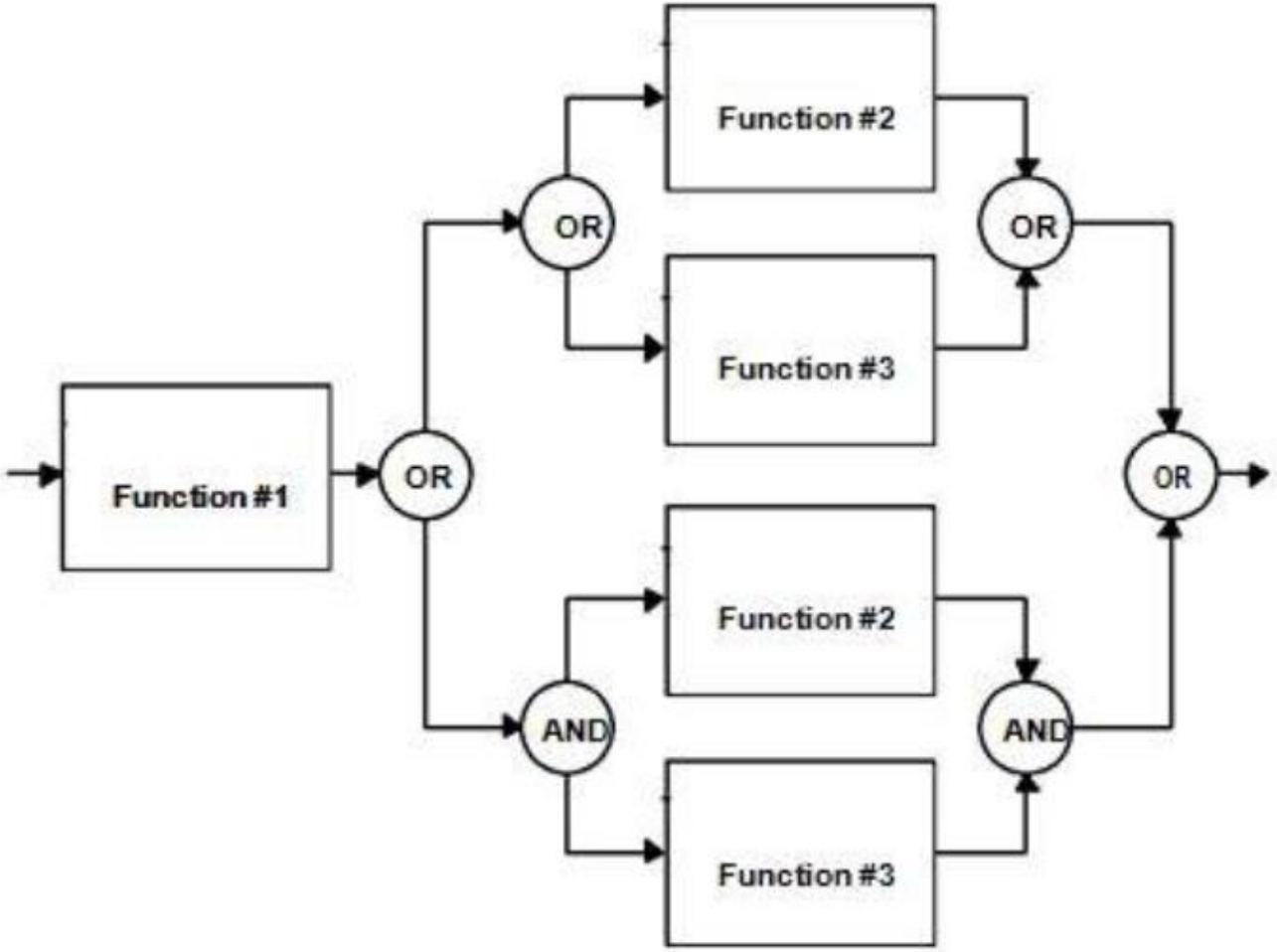
Logical Conditions: Exclusive OR (XOR)

- A select construct consists of an OR node followed by multiple branches that rejoin at a matching OR node. Each branch can contain any number of functions and control constructs.
- In contrast to a parallel construct in which all branches are executed, with a select construct only one branch is executed. Thus, the select construct is an exclusive OR.



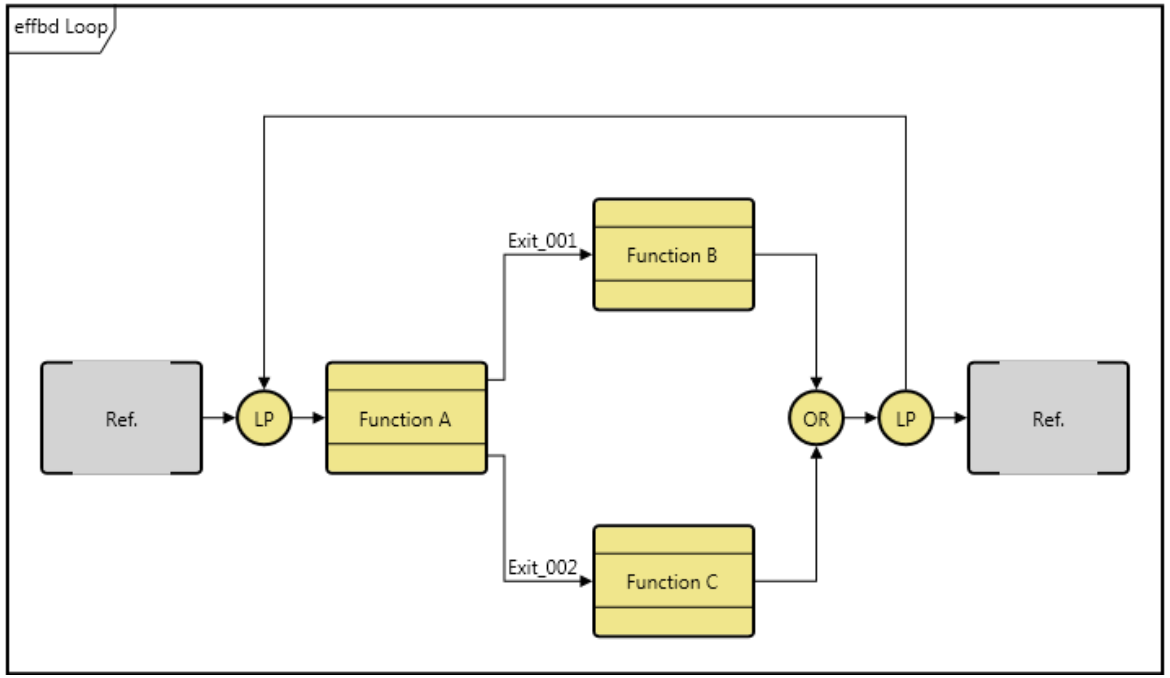
Logical conditions: or inclusive

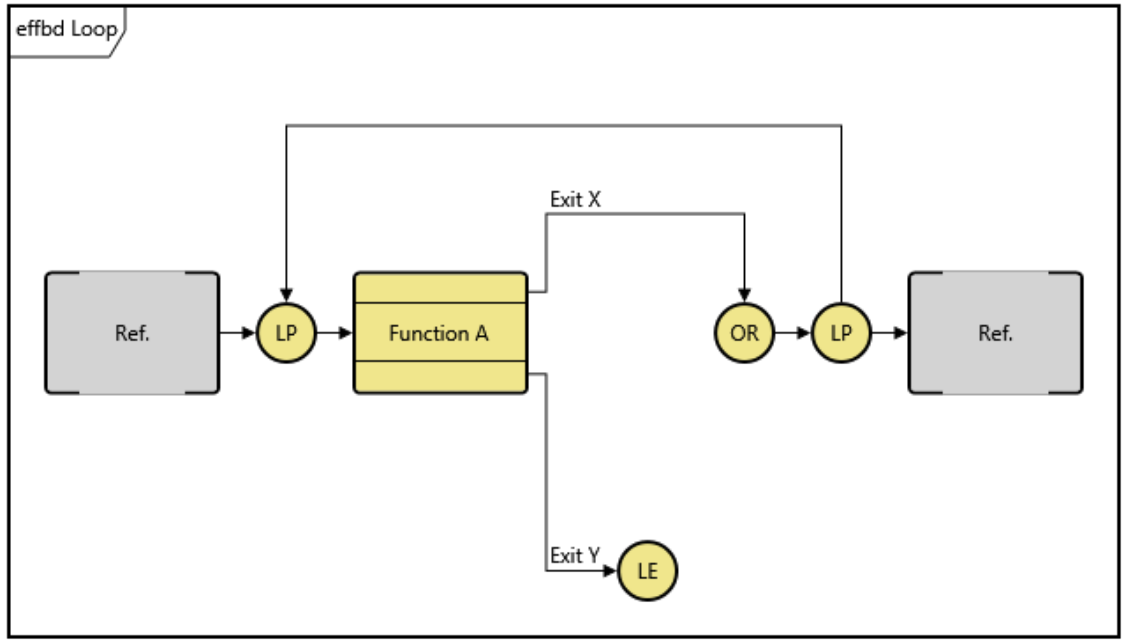
- Inclusive OR: A condition in which one, some, or all the preceding or successive multiple paths are required.

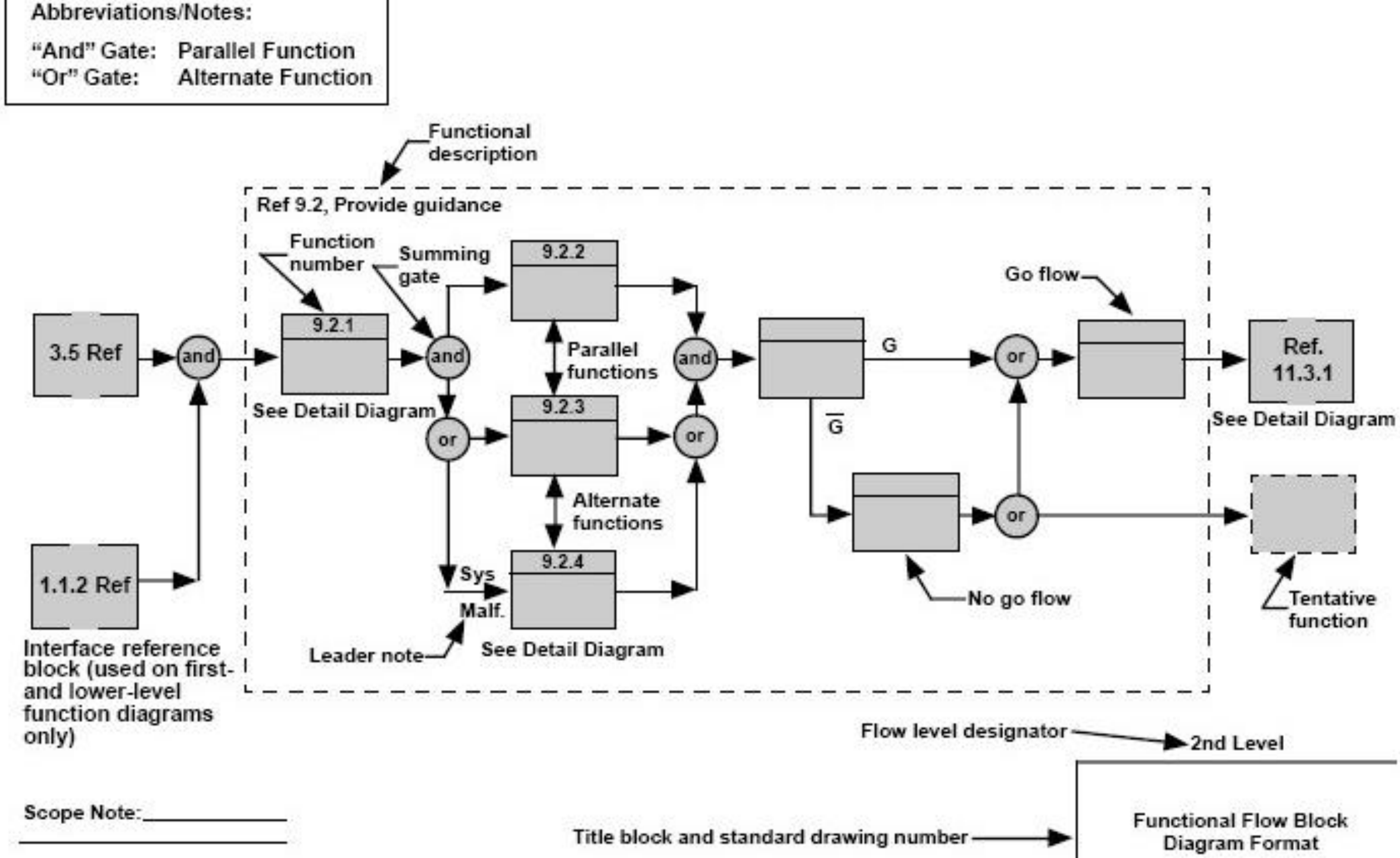


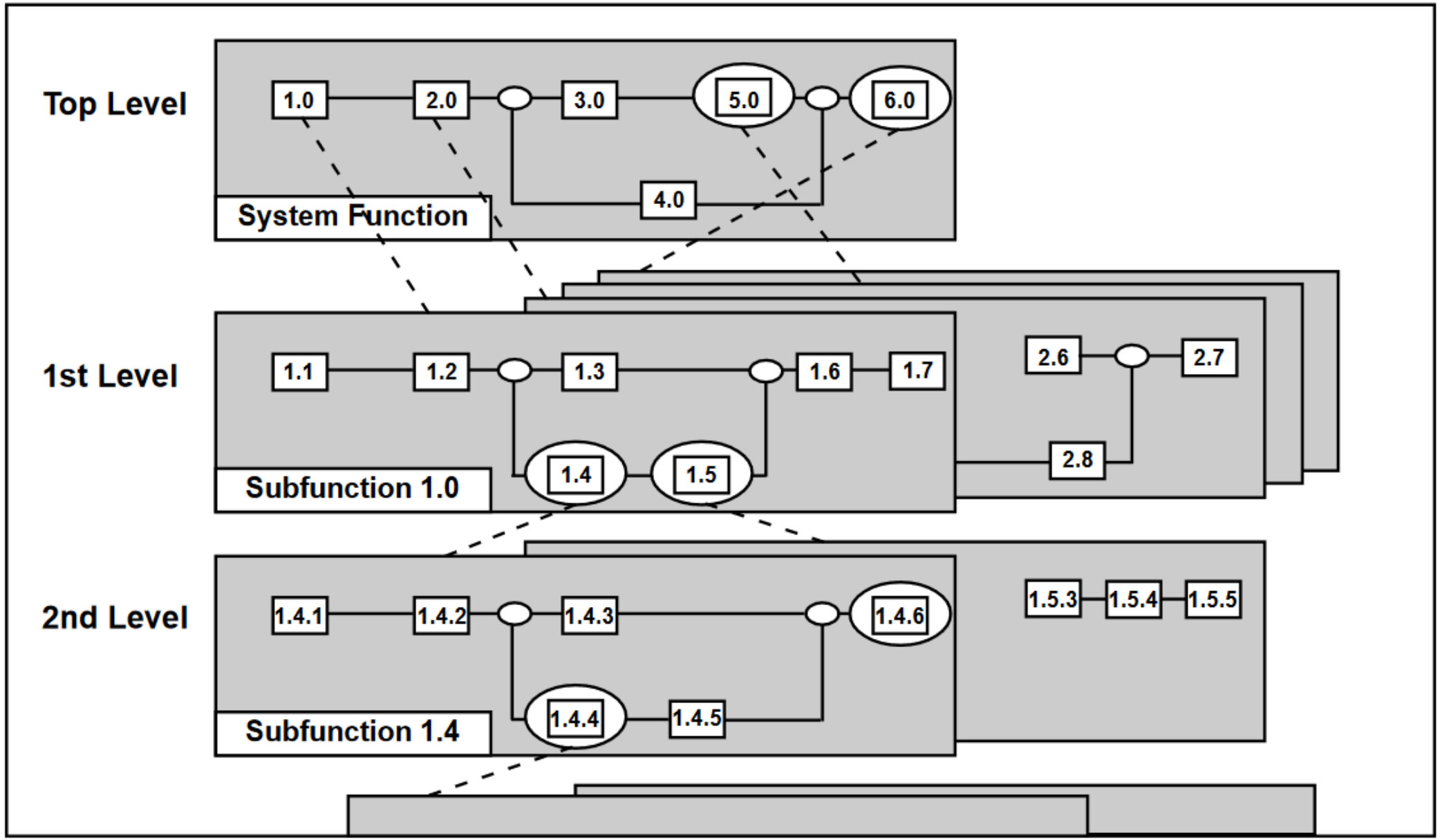
Loop

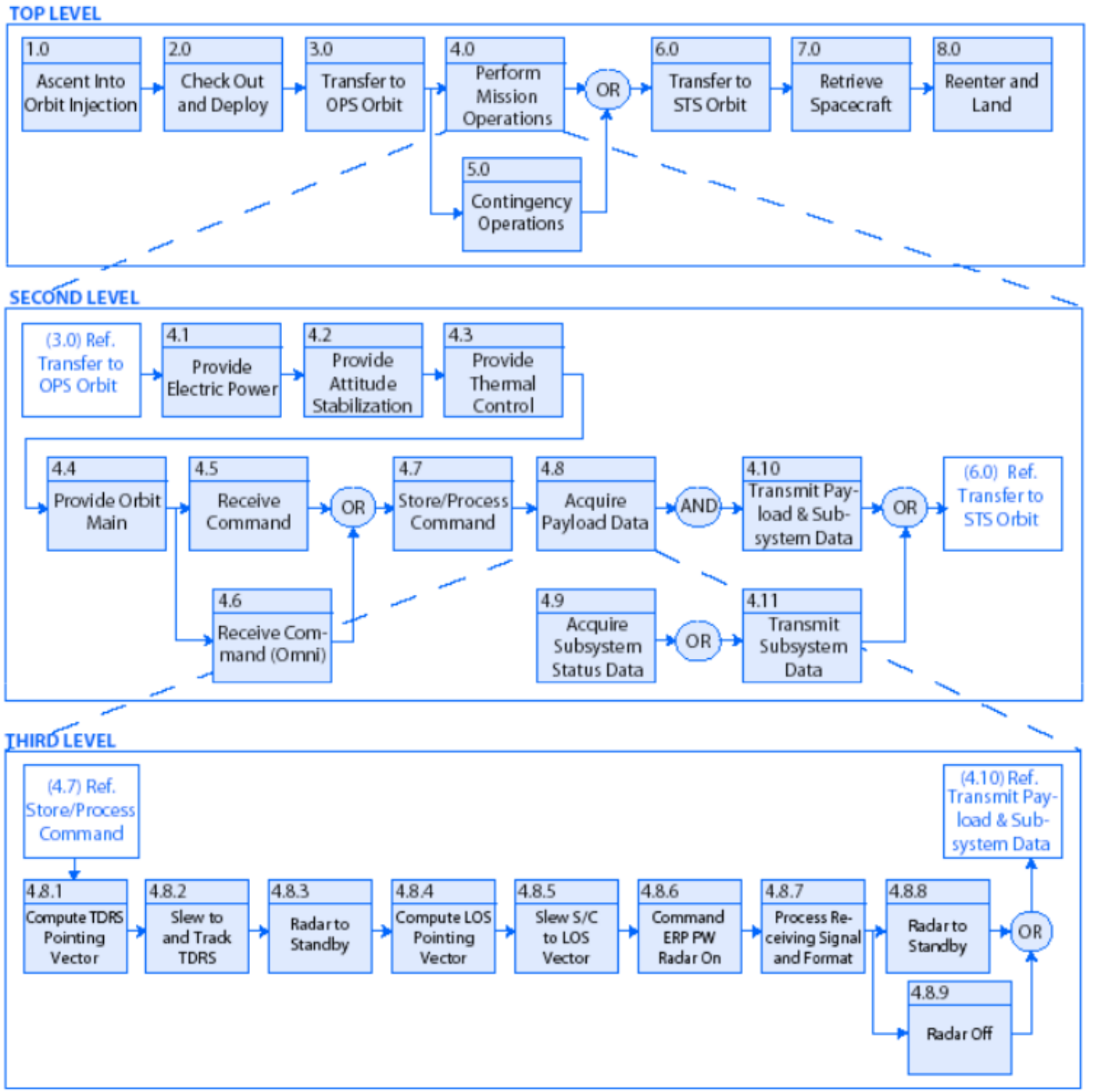
- A loop construct consists of a pair of LP nodes that enclose a branch and are connected with a loop back line. The branch can contain any number of functions and control constructs. These will be repeatedly executed in sequence.
- The branch will typically contain a loop exit construct to conditionally exit the loop construct. Without a loop exit, the loop construct becomes an endless loop.
- The loop exit construct provides the mechanism for exiting a loop. When the loop exit construct is encountered, the innermost loop is immediately terminated, enabling the construct or function following the loop.






Abbreviations/Notes:
"And" Gate: Parallel Function
"Or" Gate: Alternate Function
Functional description
Ref 9.2, Provide guidance
Function number
Summing gate
9.2.1
9.2.2
9.2.3
9.2.4
Parallel functions
Alternate functions
See Detail Diagram
3.5 Ref
1.1.2 Ref
and
or
Sys
Malf.
Leader note
See Detail Diagram
Interface reference block (used on first- and lower-level function diagrams only)
G
$\overline{G}$
Go flow
No go flow
Ref. 11.3.1
See Detail Diagram
Tentative function
Flow level designator
2nd Level
Scope Note:
Title block and standard drawing number
Functional Flow Block Diagram Format


Top Level
1.0
2.0
3.0
5.0
6.0
4.0
System Function
1st Level
1.1
1.2
1.3
1.6
1.7
2.6
2.7
2.8
1.4
1.5
Subfunction 1.0
2nd Level
1.4.1
1.4.2
1.4.3
1.4.6
1.5.3
1.5.4
1.5.5
1.4.4
1.4.5
Subfunction 1.4


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 Pay-load & Sub-system Data
OR
(6.0) Ref. Transfer to STS Orbit
4.6
Receive Com-mand (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 Pay-load & Sub-system 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 Re-ceiving Signal and Format
4.8.8
Radar to Standby
OR
4.8.9
Radar Off

enhanced Functional Flow Block Diagram

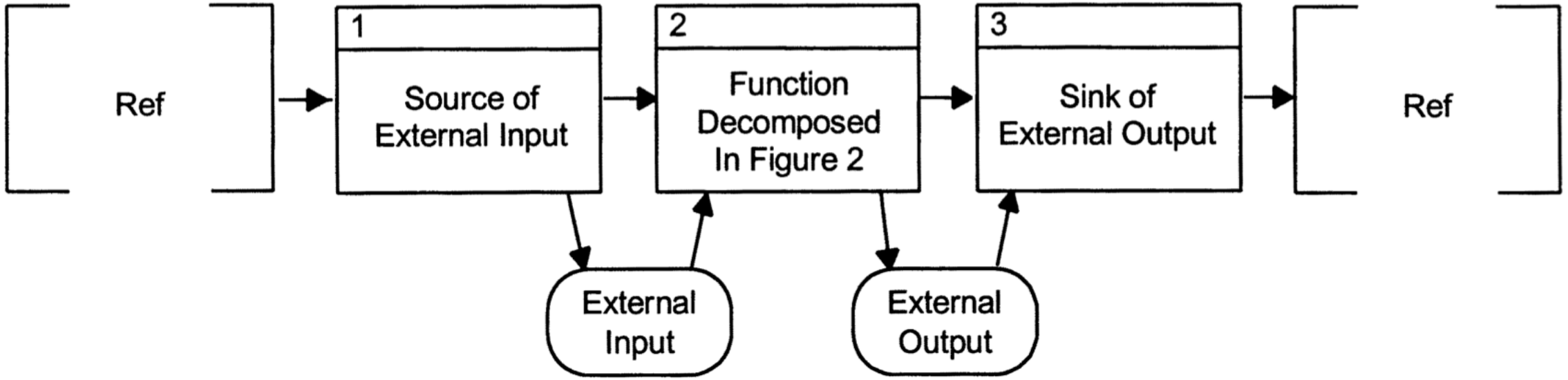


Figure 1. Enhanced functional flow block diagram context for Figure 2.

https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=822057

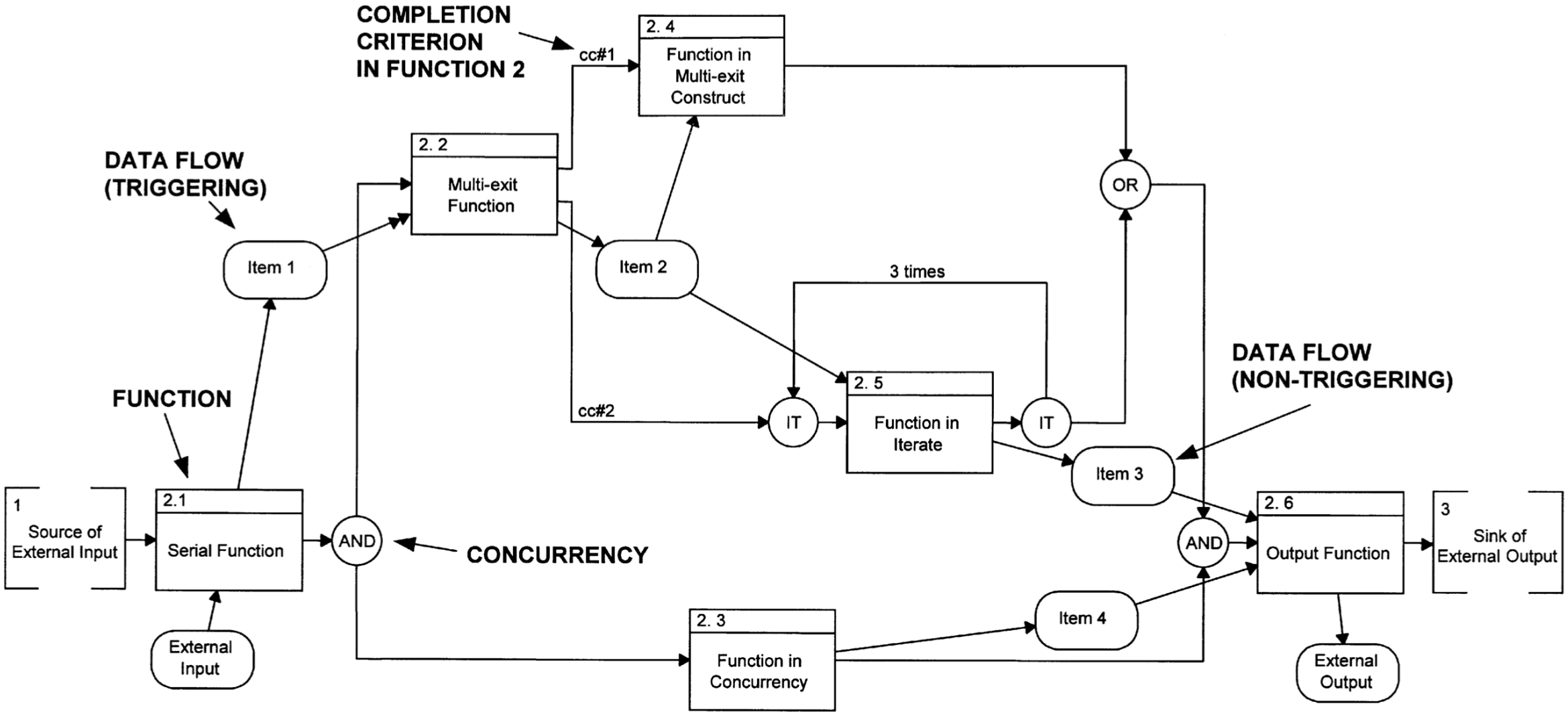
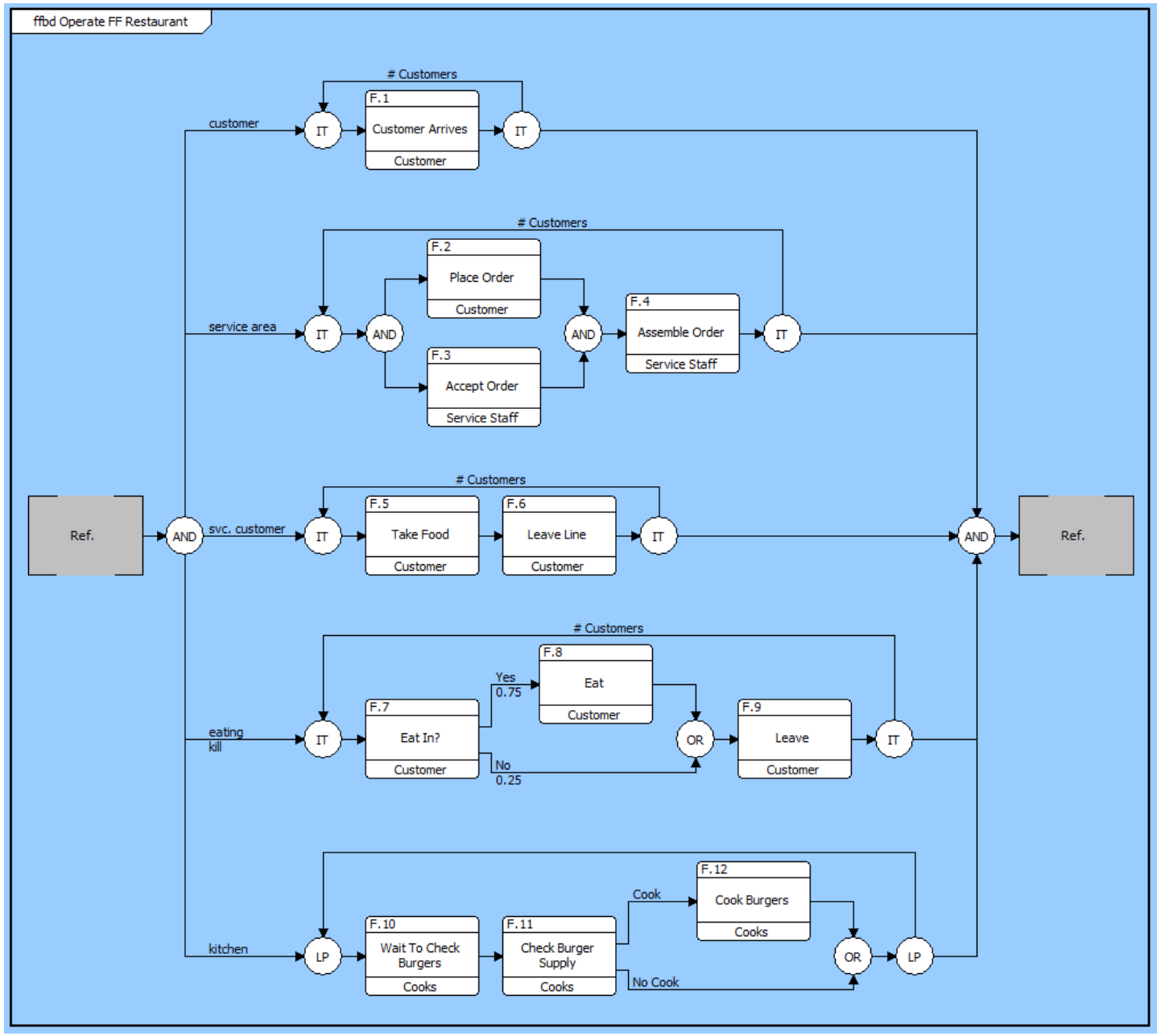


Figure 2. Enhanced functional flow block diagram for function 2 in Figure 1.



https://www.youtube.com/watch?v=oeVh9Gd8F_8

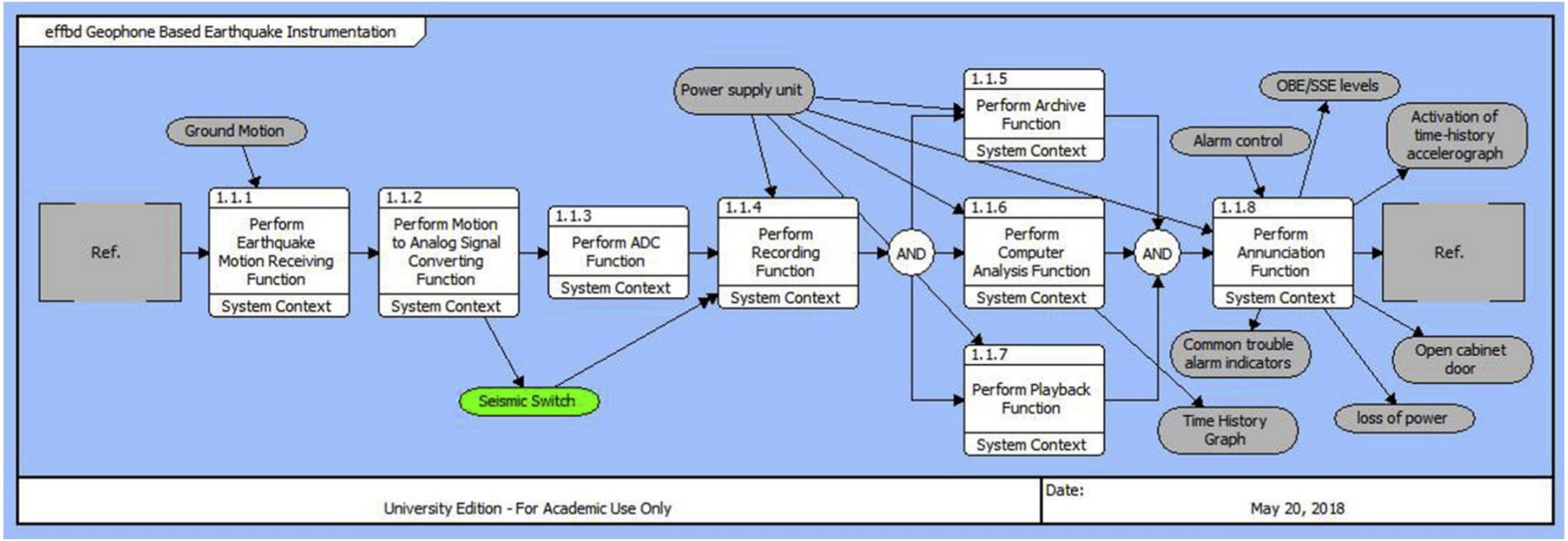


Fig. 5. EFFBD of the existing earthquake instrumentation.
EFFBD, enhanced functional flow block diagram.

Some possible analysis

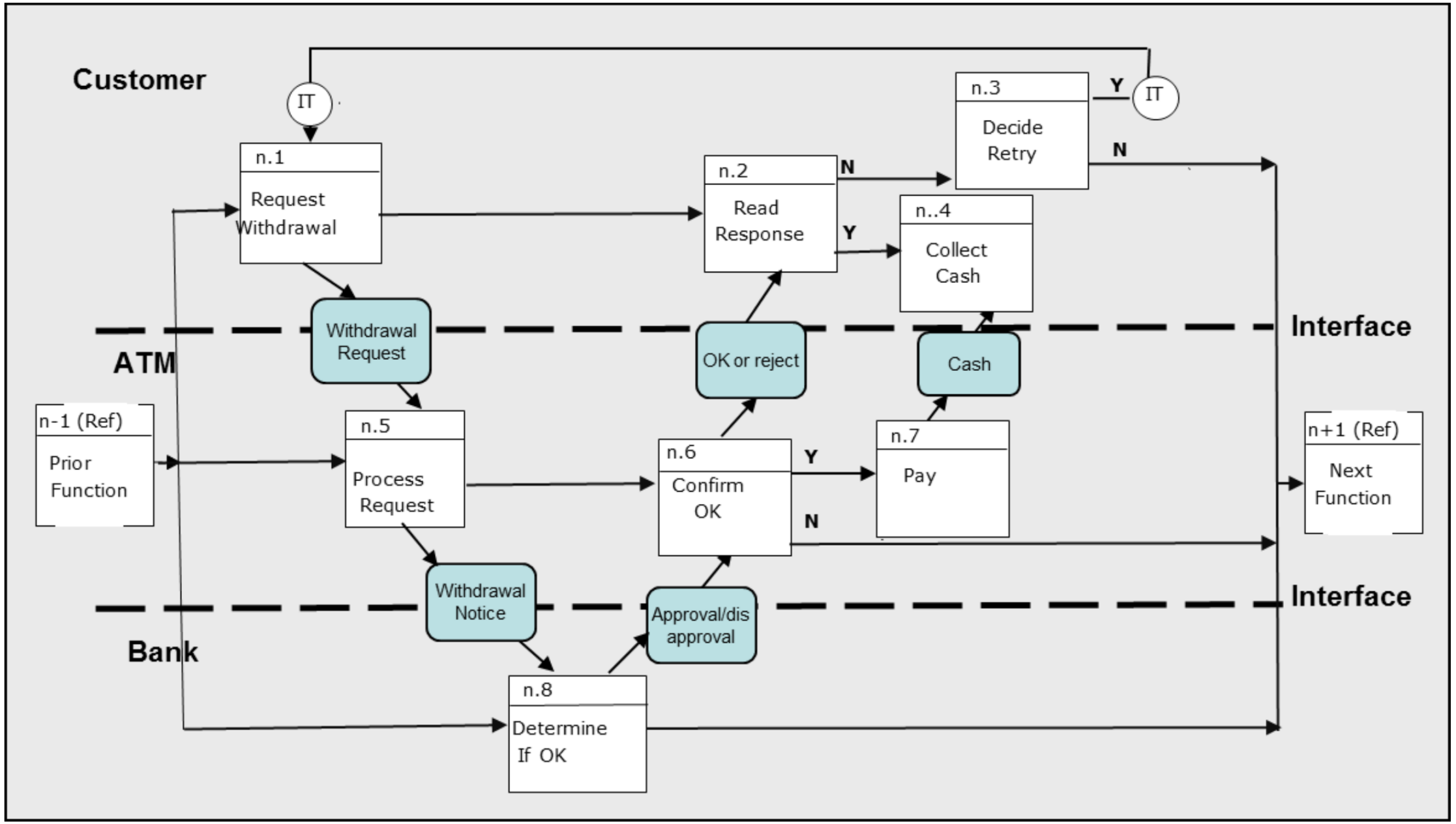
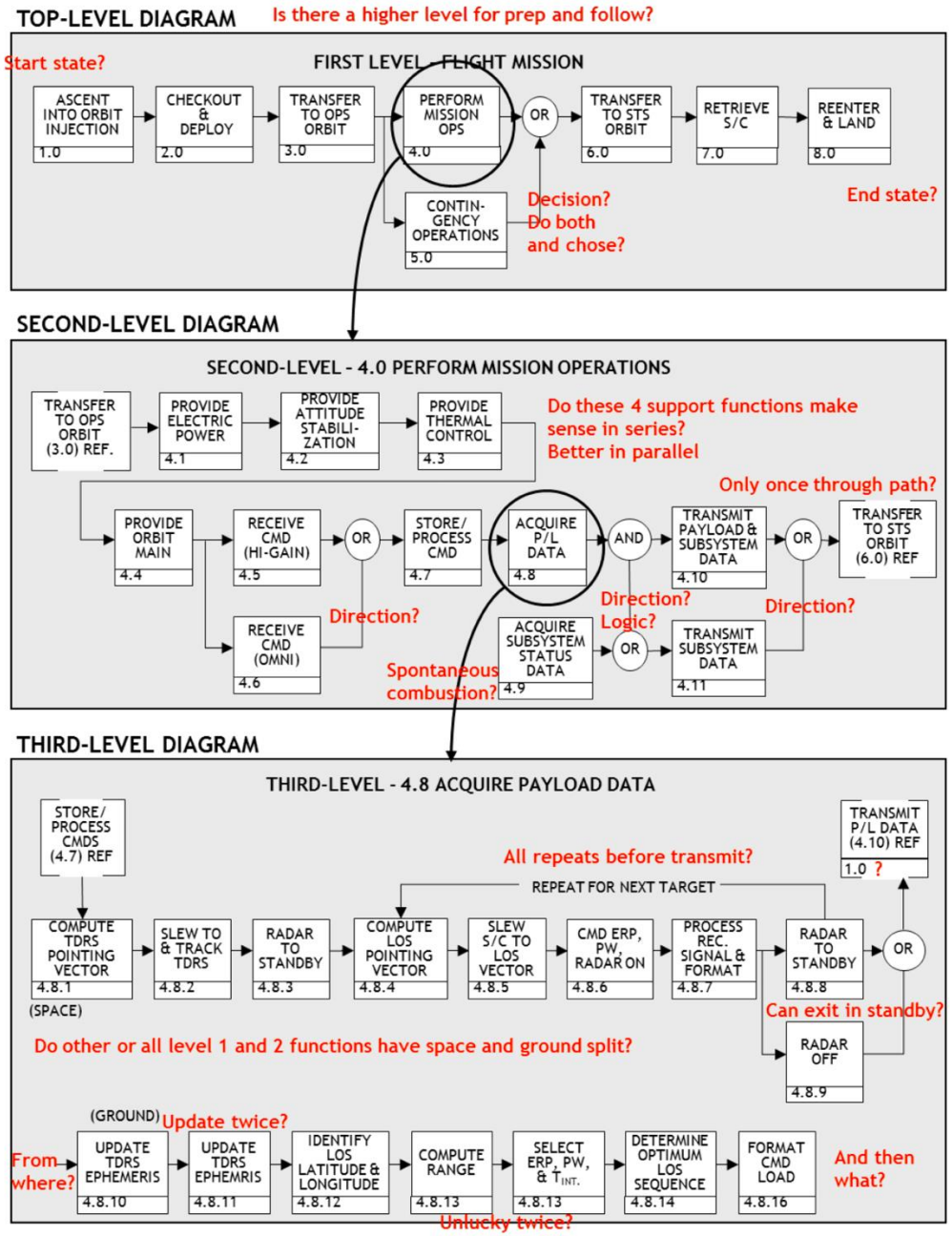


Figure 6. Example of Enhanced Functional Flow Block Diagram with Swimlanes for Allocations



Source: DSMC SE Management Guide (Jan 1990).

Figure 13. FFBD Example with Class Discussion Notes

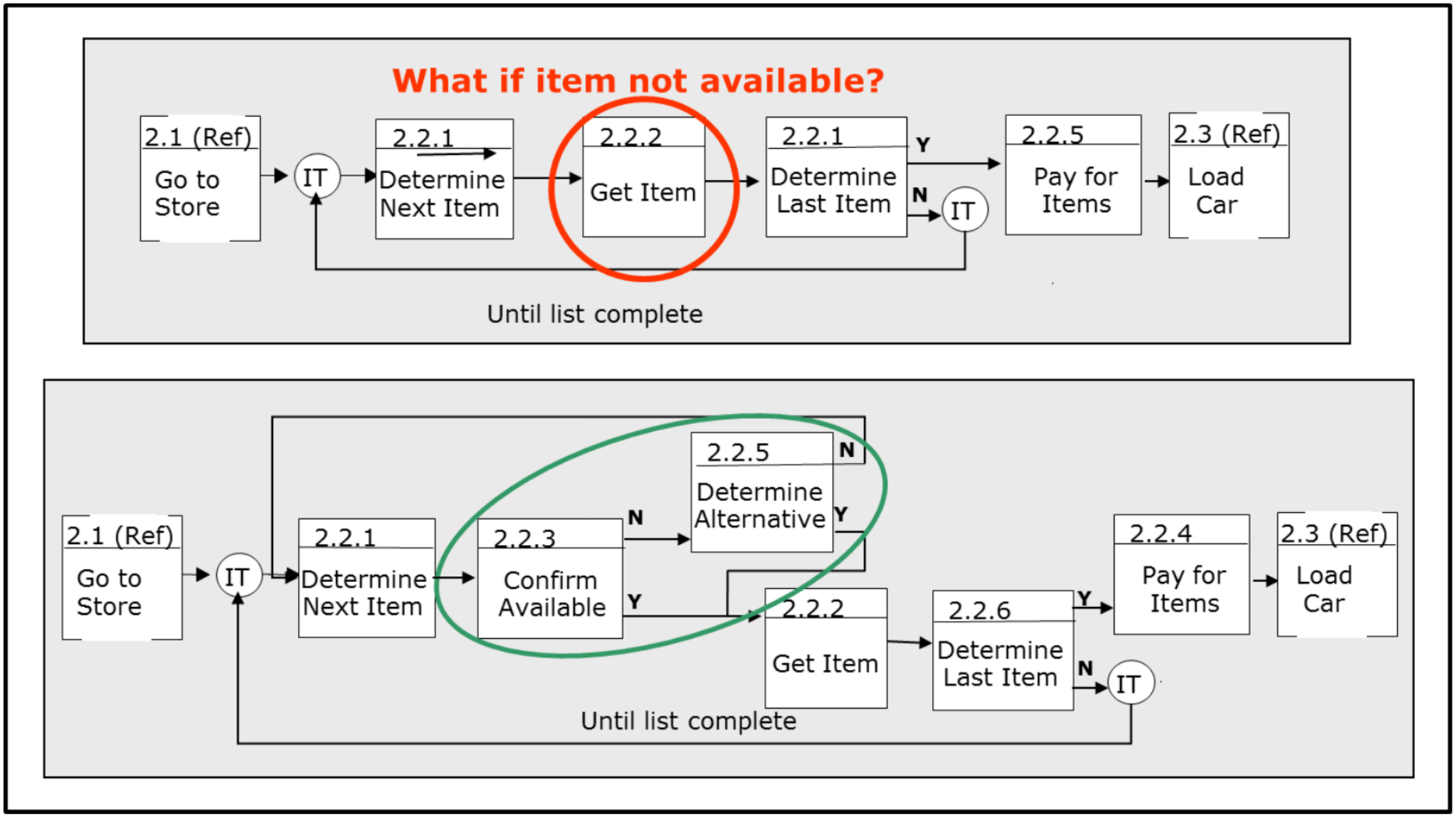
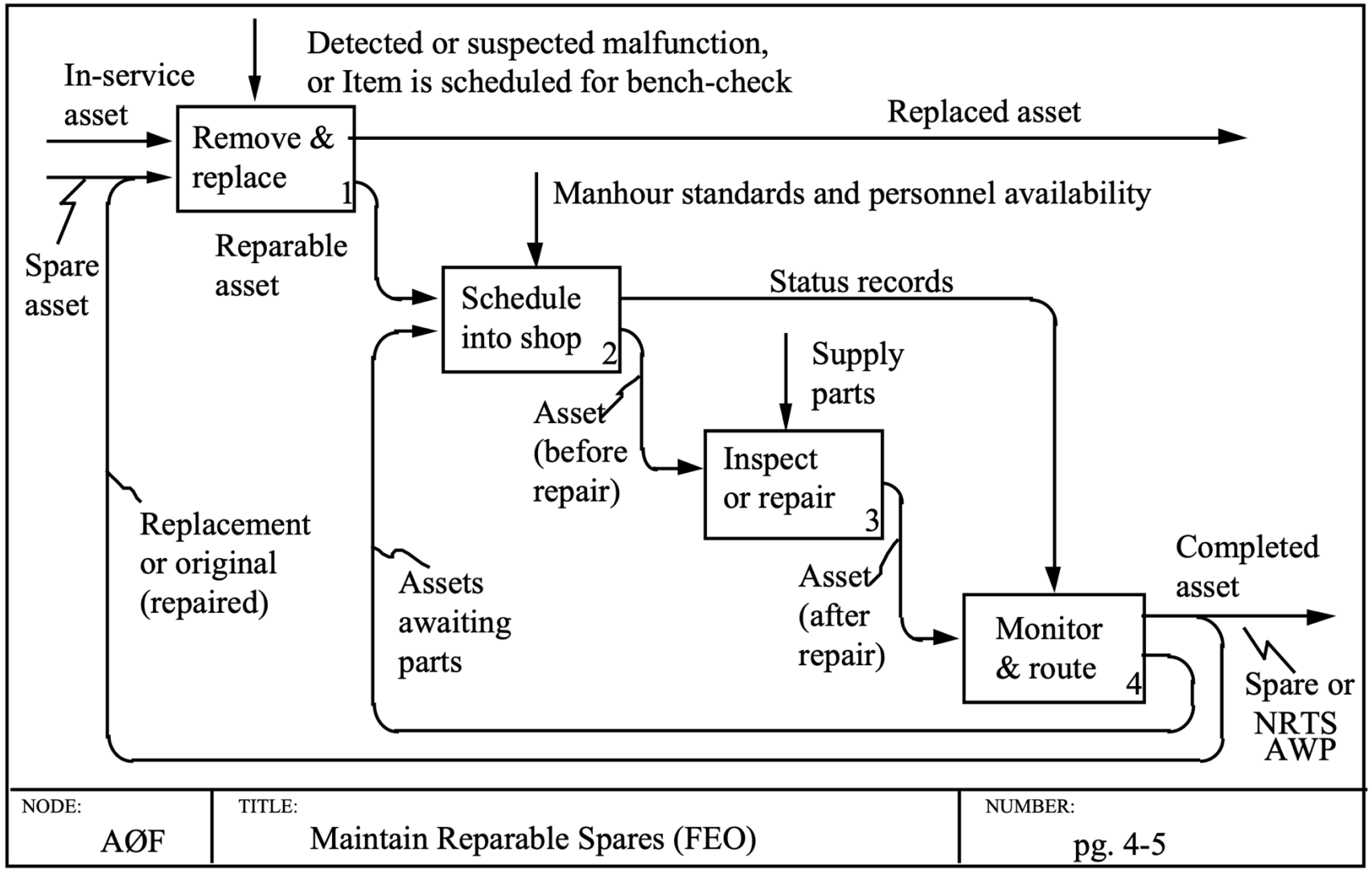


Figure 11. Functional Failure Analysis Example

IDEF0

Idef0



https://www.idef.com

Integration Definition for Function Modeling (IDEF0)

- **IDEF0** is associated with the methodology established by Douglas T. Ross in 1974.
- In parallel, the U.S. Air Force's Integrated Computer-Aided Manufacturing (ICAM) program adopted the approach and mandated a definition for the public domain called IDEF0 (1975).
-
- **IDEF0** ("Icam DEFinition for Function Modeling"), is a function modeling methodology for describing manufacturing functions, which offers a functional modeling language for the analysis, development, reengineering, and integration of information systems; business processes; or software engineering analysis.

- **IDEF0** is part of the IDEF family of modeling languages in the field of software engineering and is built on top of the **Structured Analysis and Design Technique (SADT)**.

Benefits resulting from the use of IDEF0

- **Identifies needs**: A good IDEF0 template will look very simple and easy to understand. Needs and opportunities for improvement revealed during the modeling effort may seem obvious at first, but would have gone unnoticed otherwise.

- **Builds consensus**: The diversity of knowledge, training, skills, and expertise of a team can hinder communication. The reader/author model's critique method can establish a well-defined common ground for understanding by the entire team.

Benefits resulting from the use of IDEF0

- **Improves vision**: The condensed graphic image provided by the model presents a thought-provoking baseline on which to consider improvements. An analyst or corporate planner who considers the potential use of a new technology or new method in light of this baseline can identify specific application opportunities.

- **Provides a foundation for an open architecture**: The model structure can be used to define the interfaces between system elements and to identify precise interfaces for defining an open systems architecture. The model narrows the scope precisely, showing where the modeled system fits into the larger picture.

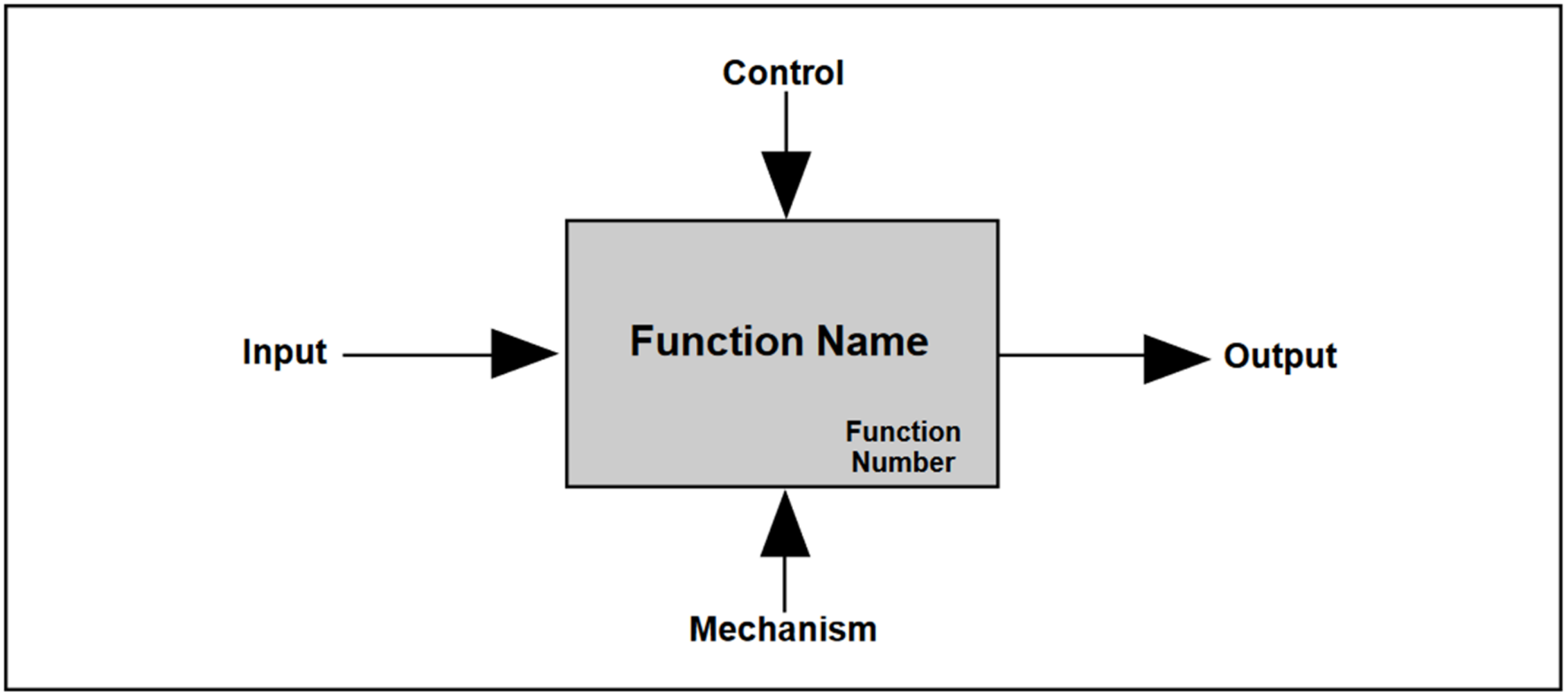
Benefícios resultantes do uso do IDEF0

- **Suporta o controle de gerenciamento por meio de métricas**: O modelo pode servir como uma linha de base para custo, tempo, capacidade de fluxo e outras métricas relevantes para o projeto de um novo sistema. Se os detalhes do modelo não forem suficientes para anexar métricas específicas, a falta de detalhes deve ser usada para identificar onde é necessária mais decomposição. Detalhes mais finos na modelagem facilitarão uma medição mais precisa.

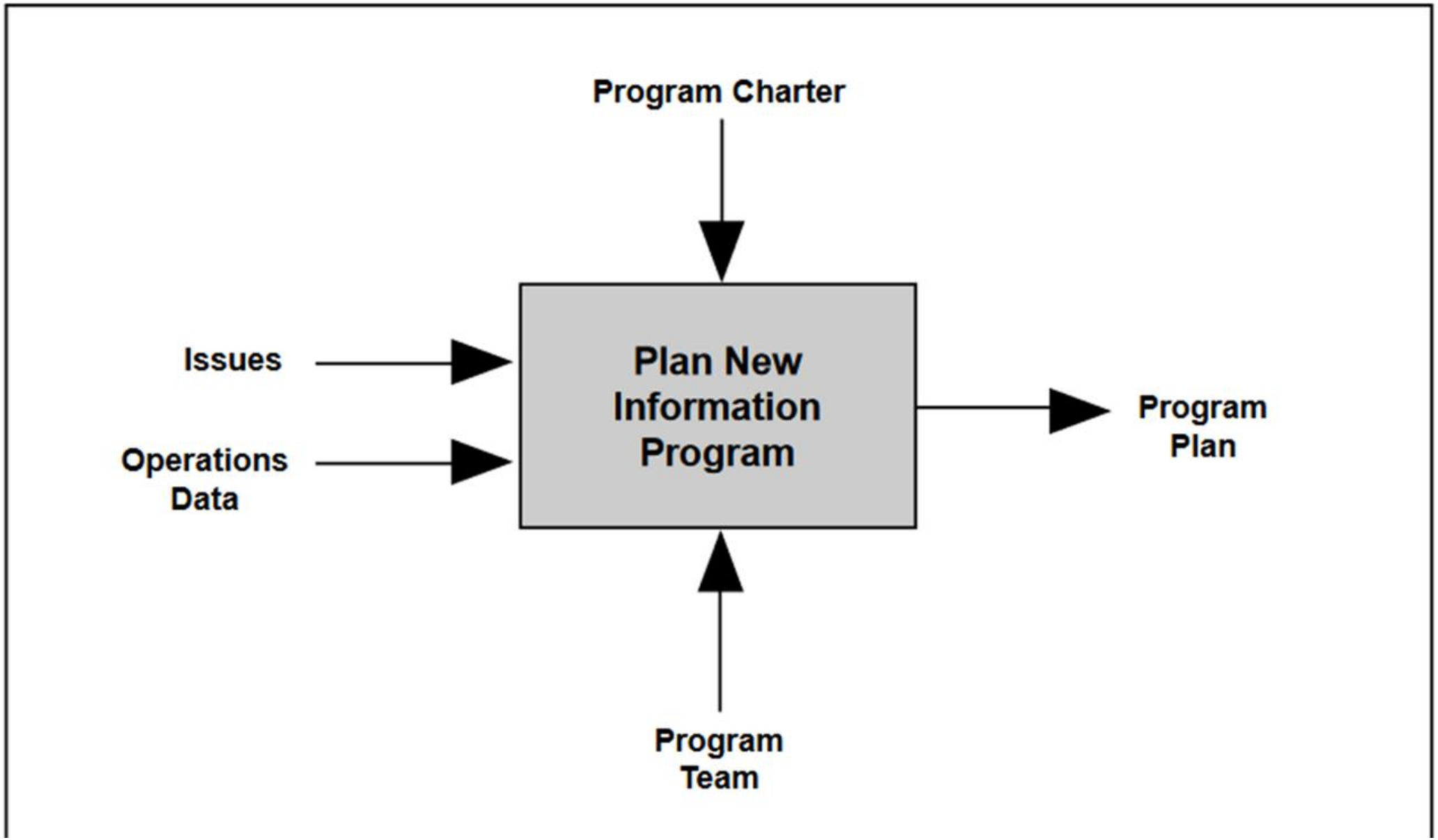
- **Define variantes para uso mais amplo do sistema de suporte***:* Os sistemas de suporte que são aplicados a várias atividades no modelo podem se beneficiar de uma análise cuidadosa da funcionalidade real necessária em cada ponto de uso. Variantes e versões do sistema de suporte podem economizar custos consideráveis se o sistema de suporte puder ser ajustado para cada ponto de uso específico.

The position at which the arrow connects matter

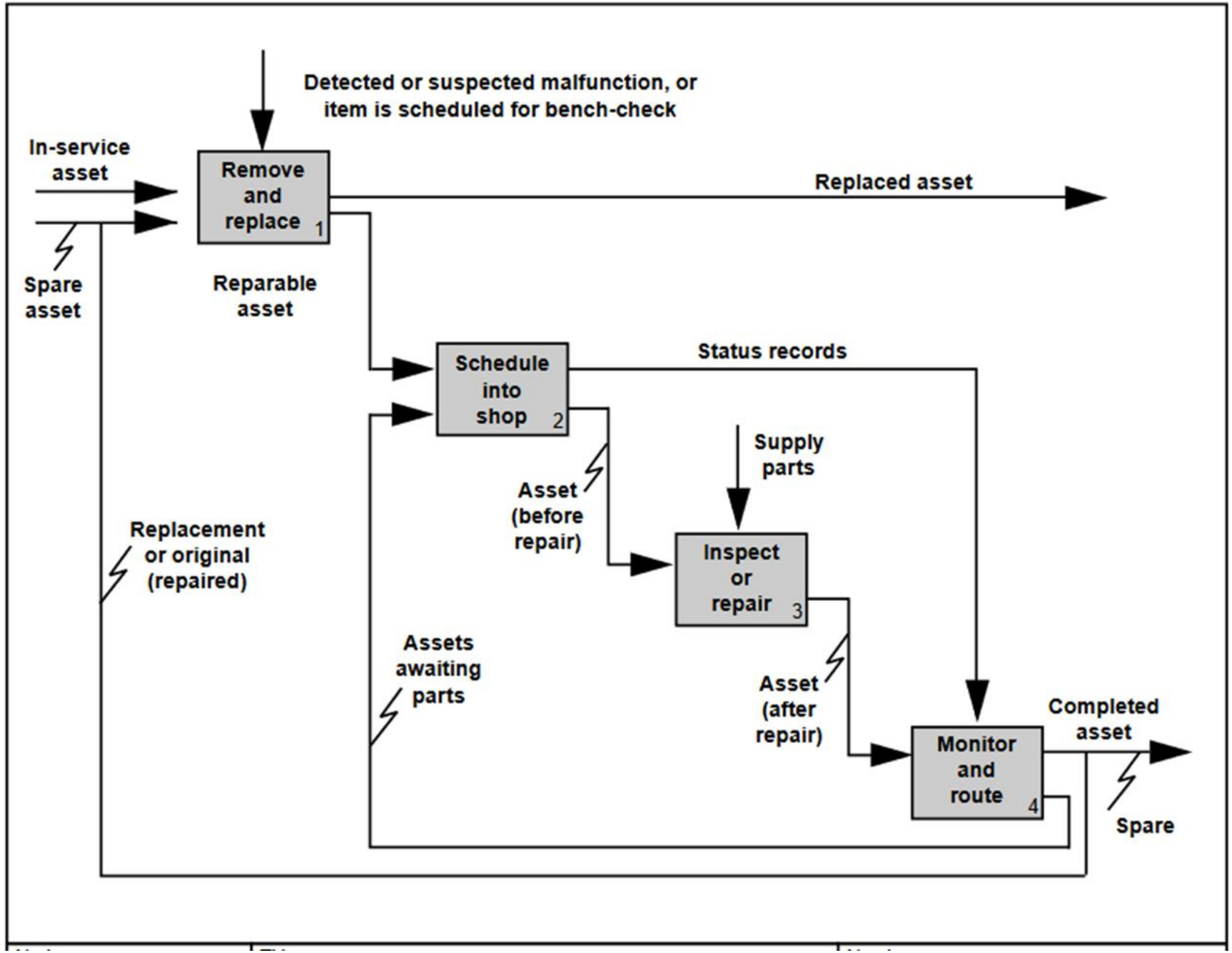
- **Controls** go into the top of the box.
- The **inputs**, data, or objects actuated by the operation, type the box from the left.
- The **outputs** come out of the right side of the box.
- The arrows of the **mechanism** that provide means of support to perform the function join (point towards) the bottom of the Box.



DoD. Systems Engineering Fundamentals. 2001.



DoD. Systems Engineering Fundamentals. 2001.



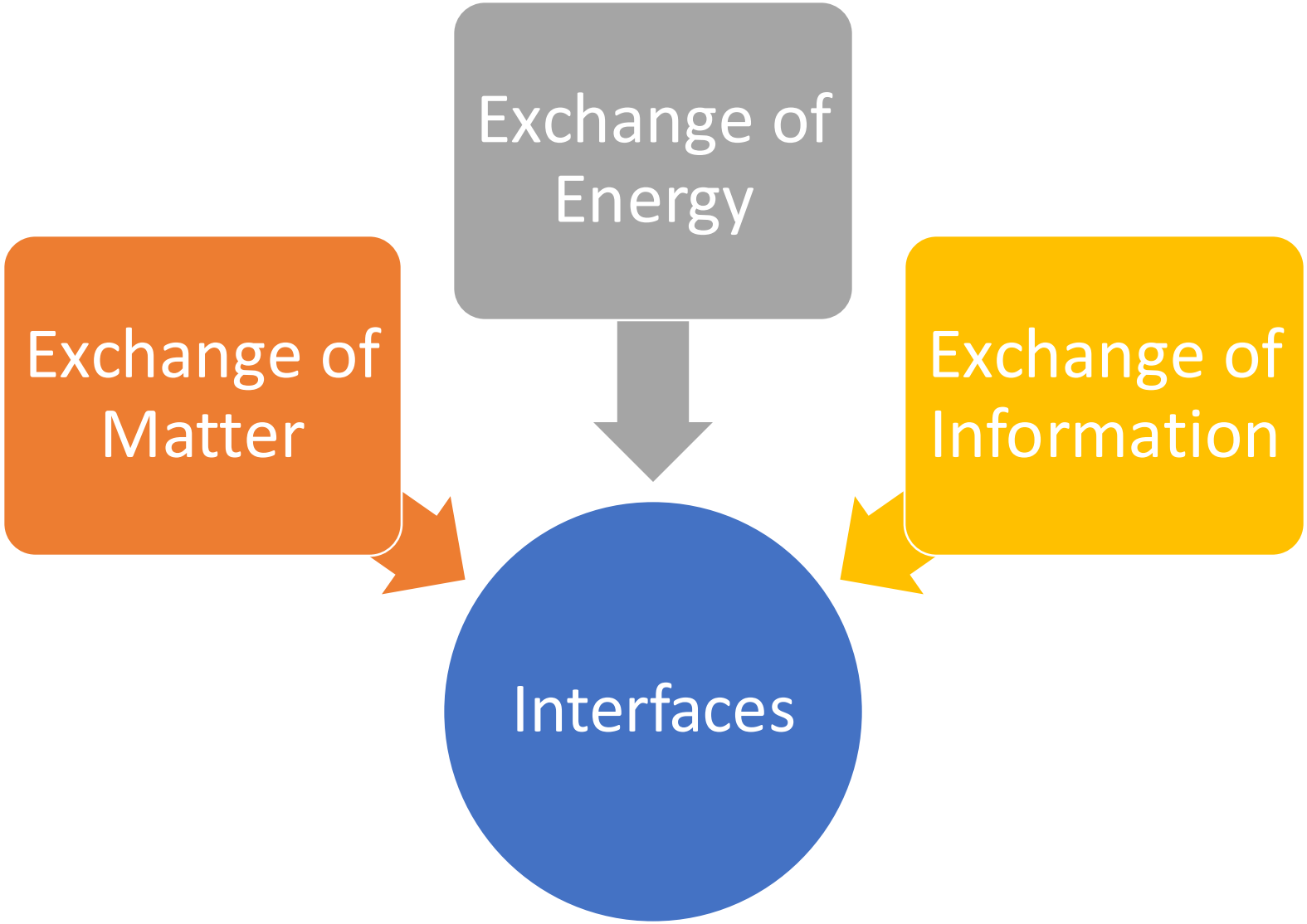
DoD. Systems Engineering Fundamentals. 2001.

N2

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**.

Common Interfaces



Other possible interface domains

Data types	Representation	Applications
Component-based (ProdBuct)	Component relationships	System architecting, engineering and design
People-based (Organization)	Organizational unit relationships	Organizational design, interface management, team integration
Activity-based (Process)	Activity input/output relationships	Process improvement, project scheduling, iteration management, information flow management
Parameter-based (low-level Process)	Design parameter relationships	Low level activity sequencing and process construction, sequencing design decisions

N2 diagrams

https://www.nasa.gov/sites/default/files/atoms/files/nasa_systems_engineering_handbook.pdf

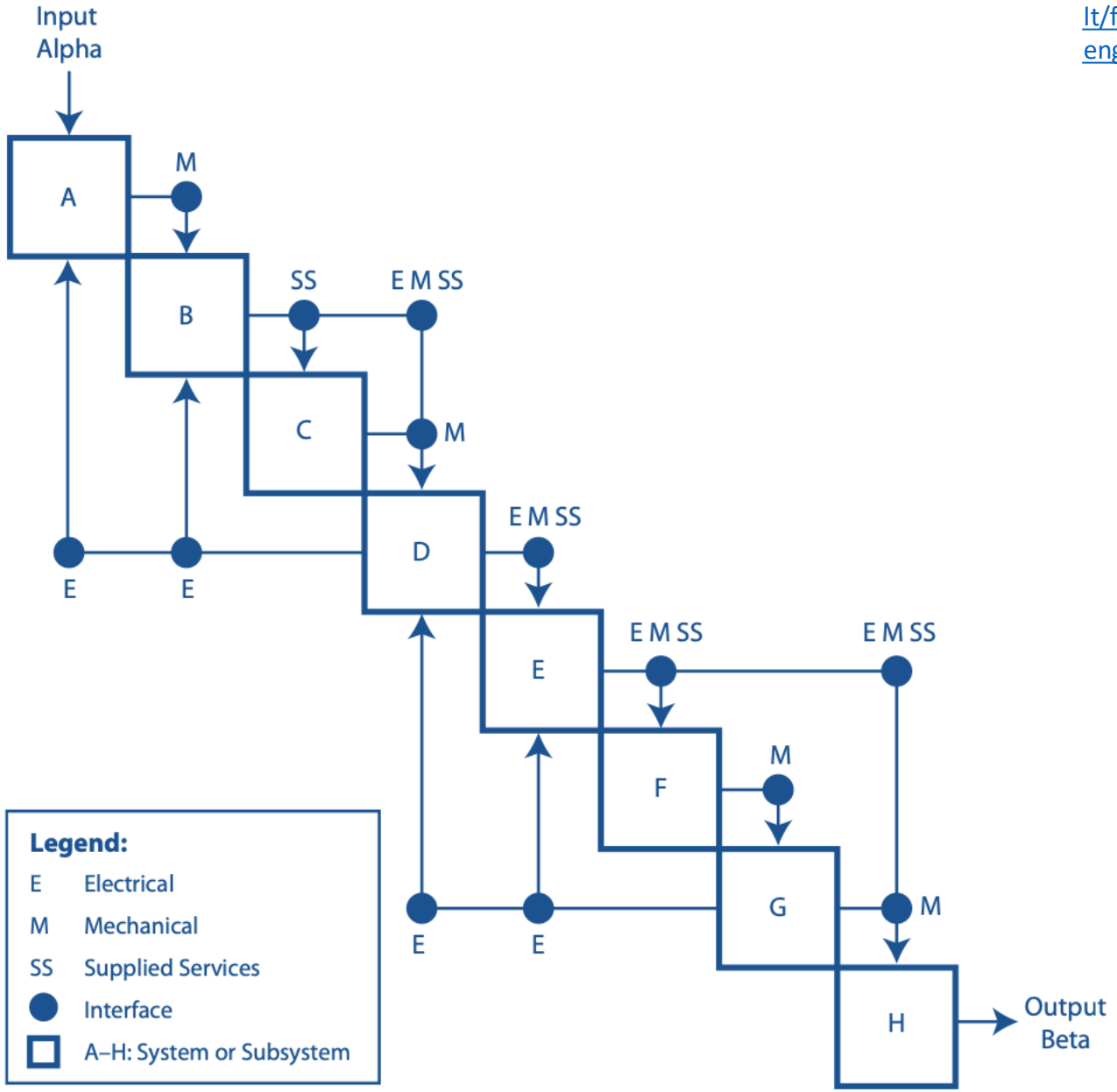


Figure 4.3-4 Example of an N2 diagram

N2 Analysis

- N2 Analysis is a tool that uses a NxN matrix to **record the interconnections between elements** of a system. It has several 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.

N2 Diagrams

- The N2 diagram has been used extensively to **develop** data **interfaces**, primarily in the software areas. However, it can also be used to develop hardware interfaces.
- The **system functions (or architecture elements) are placed on the diagonal**; the remainder of the squares in the N x N matrix represent the interface inputs and outputs. Where a blank appears, there is no interface between the respective functions.
- Data flows in a **clockwise direction between functions** (e.g., the symbol F1 F2 indicates data flowing from function F1, to function F2).
- The data being transmitted can be defined in the appropriate squares.

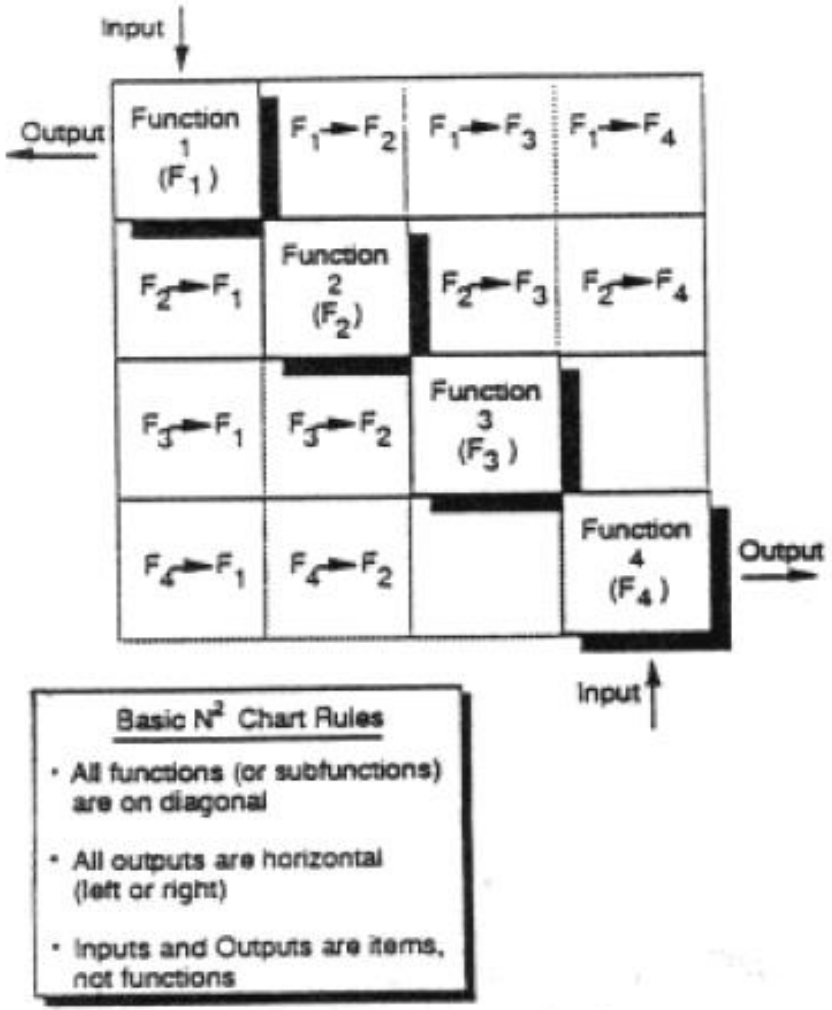
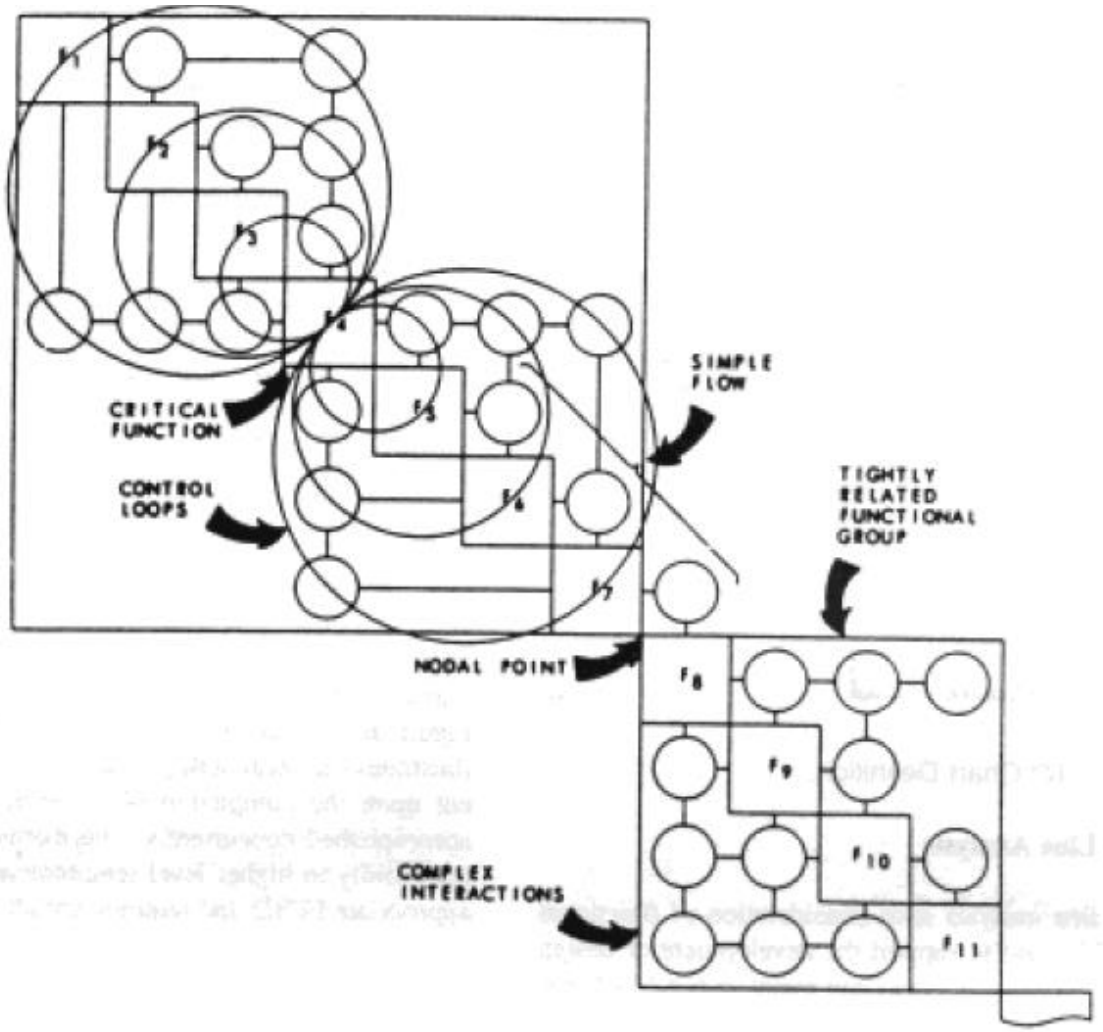


Figure B-7 — N^2 Chart Definition.

- 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.



DSM – Design Structure Matrix

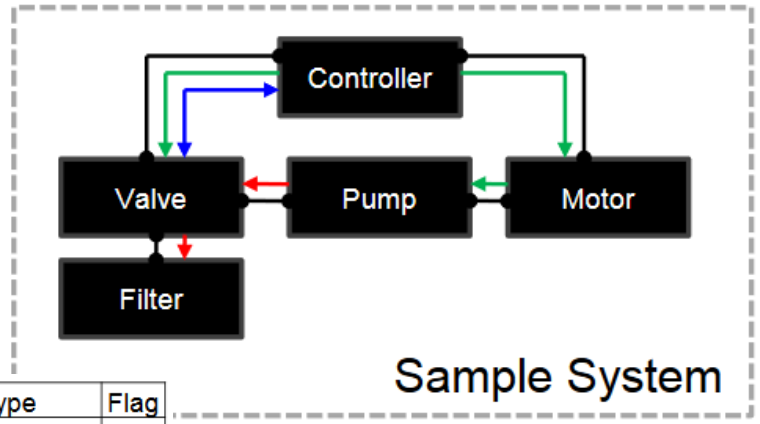
https://dsmweb.org

DSM – Design Structure Matrix

- Research on matrix-based complexity management has come a long way. Originating from a process focus with the first published formulation of a **Design Structure Matrix (DSM) by Don Steward in 1981**, a whole community has developed around this research.
- DSMs can have different qualities:
  - Binary DSMs represent only the existence of a relation,
  - whereas numerical DSMs represent a numerical value (also called “weight”) to represent the strength of a relation.
  - DSMs can either be directed or non-directed.

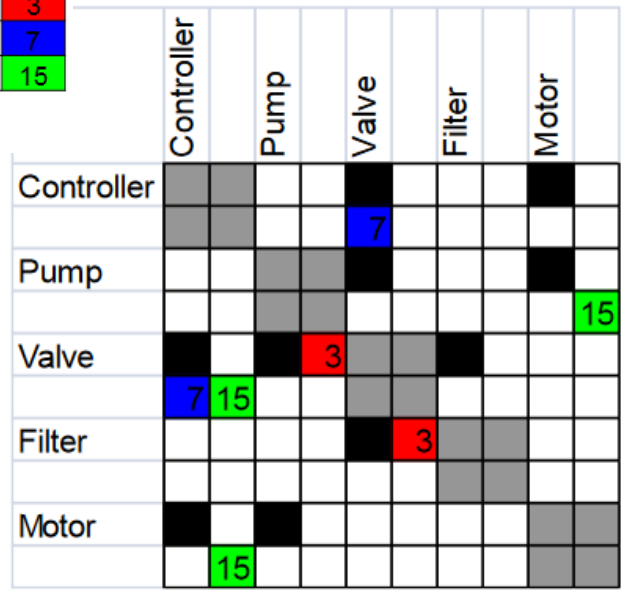
Design Structure Matrix

- **Architecture Definition**: The embodiment of concept, and the allocation of physical/informational function (process) to elements of form (objects) and definition of structural interfaces among the objects



Number	Type	Flag
0	No Connection	0
1	Mechanical	
2	Flow	3
3	Information	7
4	Energy	15

Key



DSM

- DSM captures connectivity of components => architecture
- DSM provides analysis capability not present in a traditional schematic

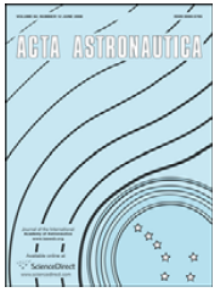
Example

Acta Astronautica 66 (2010) 937–949

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro



An application of the Design Structure Matrix to Integrated Concurrent Engineering

Mark S. Avnet *, Annalisa L. Weigel

Massachusetts Institute of Technology, Cambridge, MA, USA

Parameters exchange

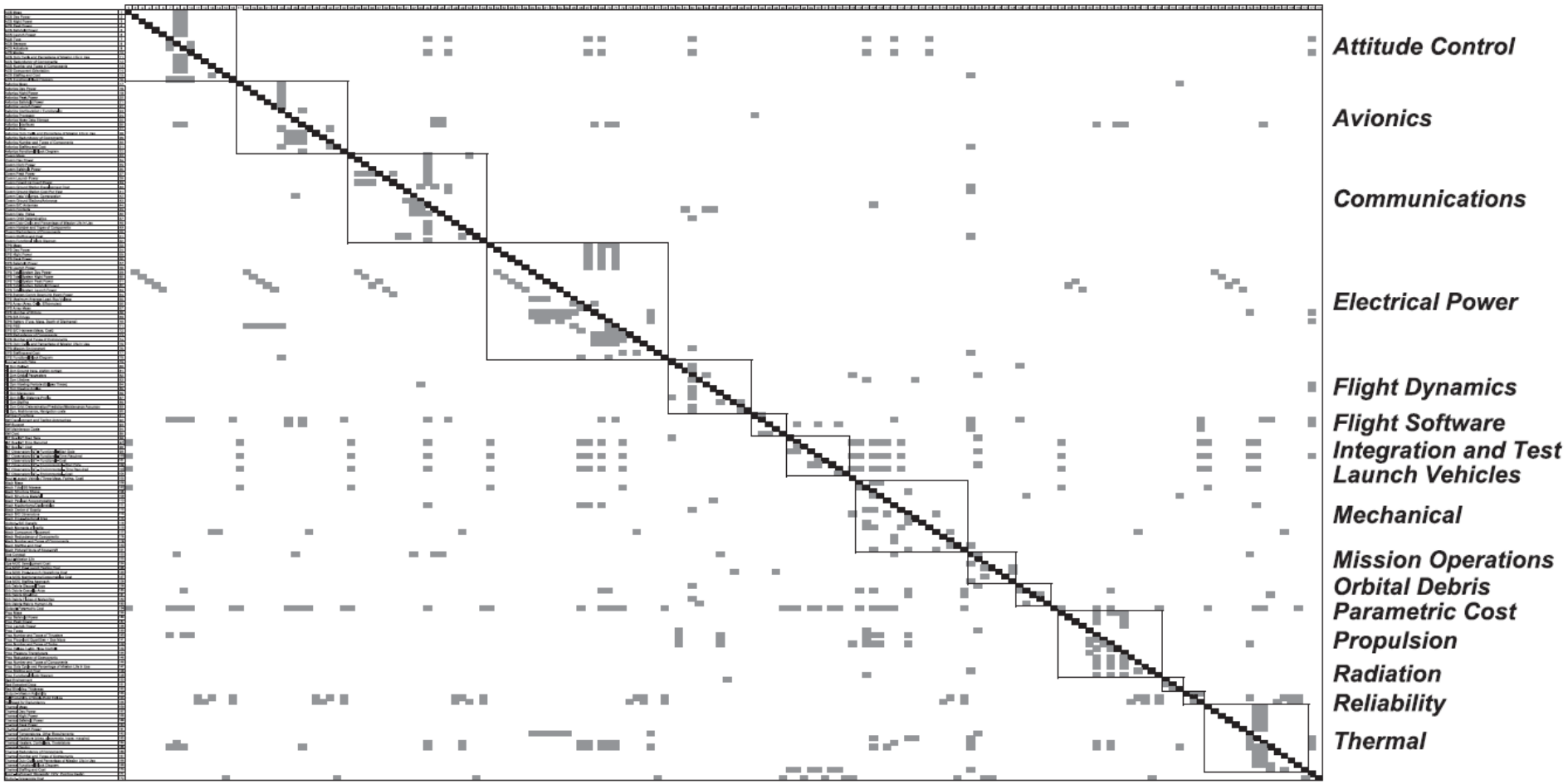


Fig. 2. Parameter-based DSM for the typical ICE design process. The DSM is organized as an alphabetical sequence of the 16 disciplines involved. The blocks along the diagonal encapsulate the work that is internal to each discipline, and the off-diagonal marks represent information flow across disciplines.

Dependencies among disciplines

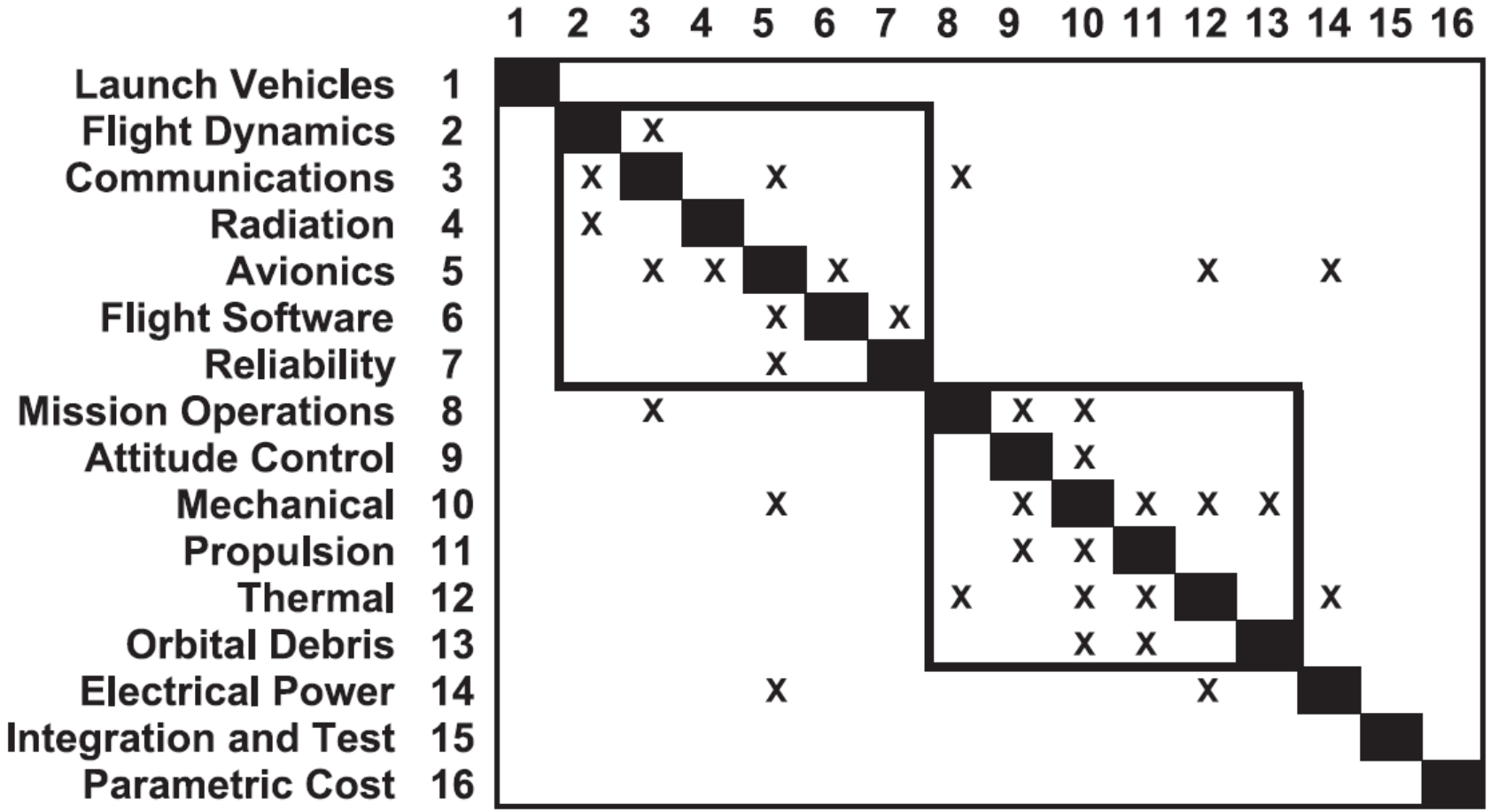
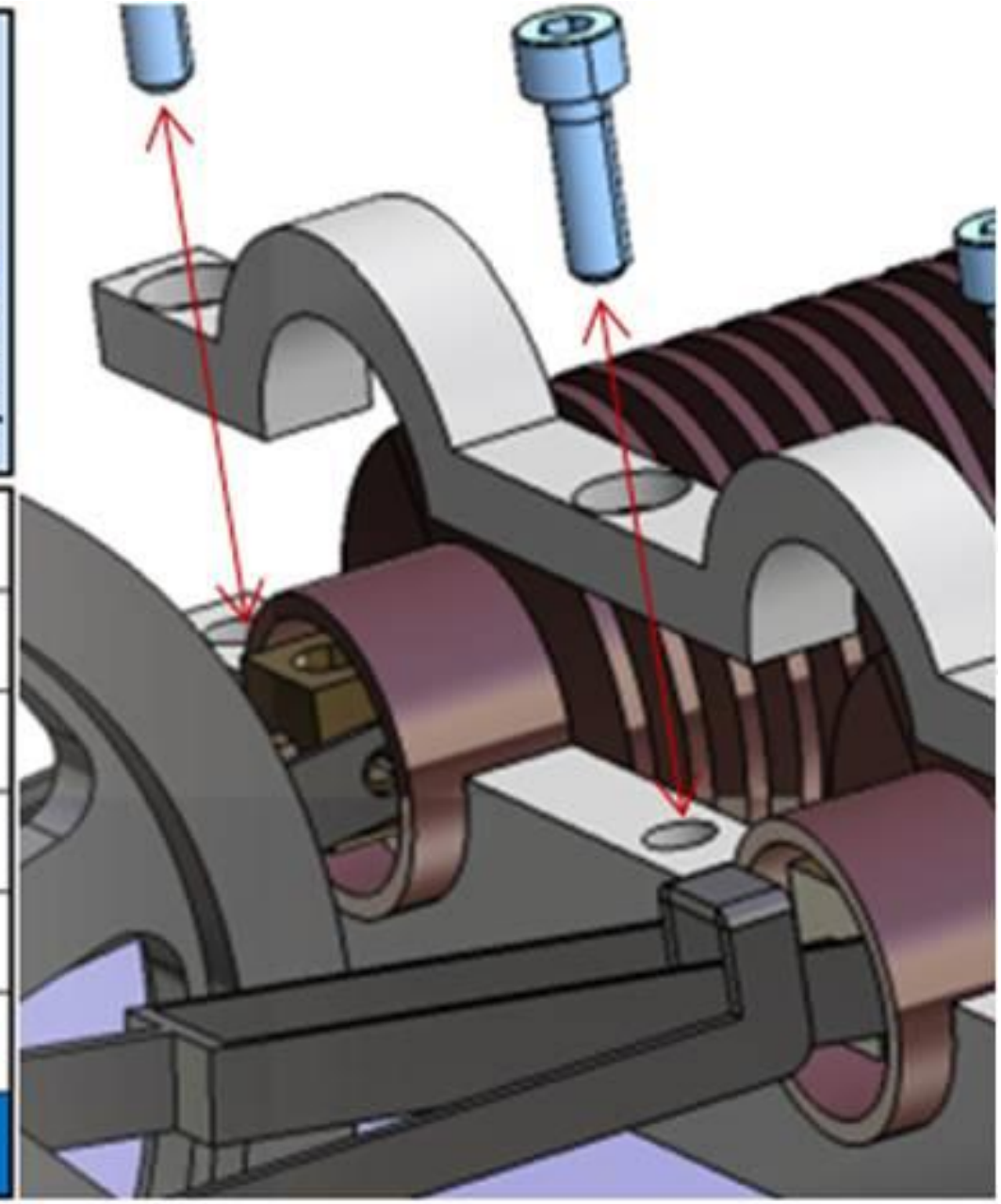
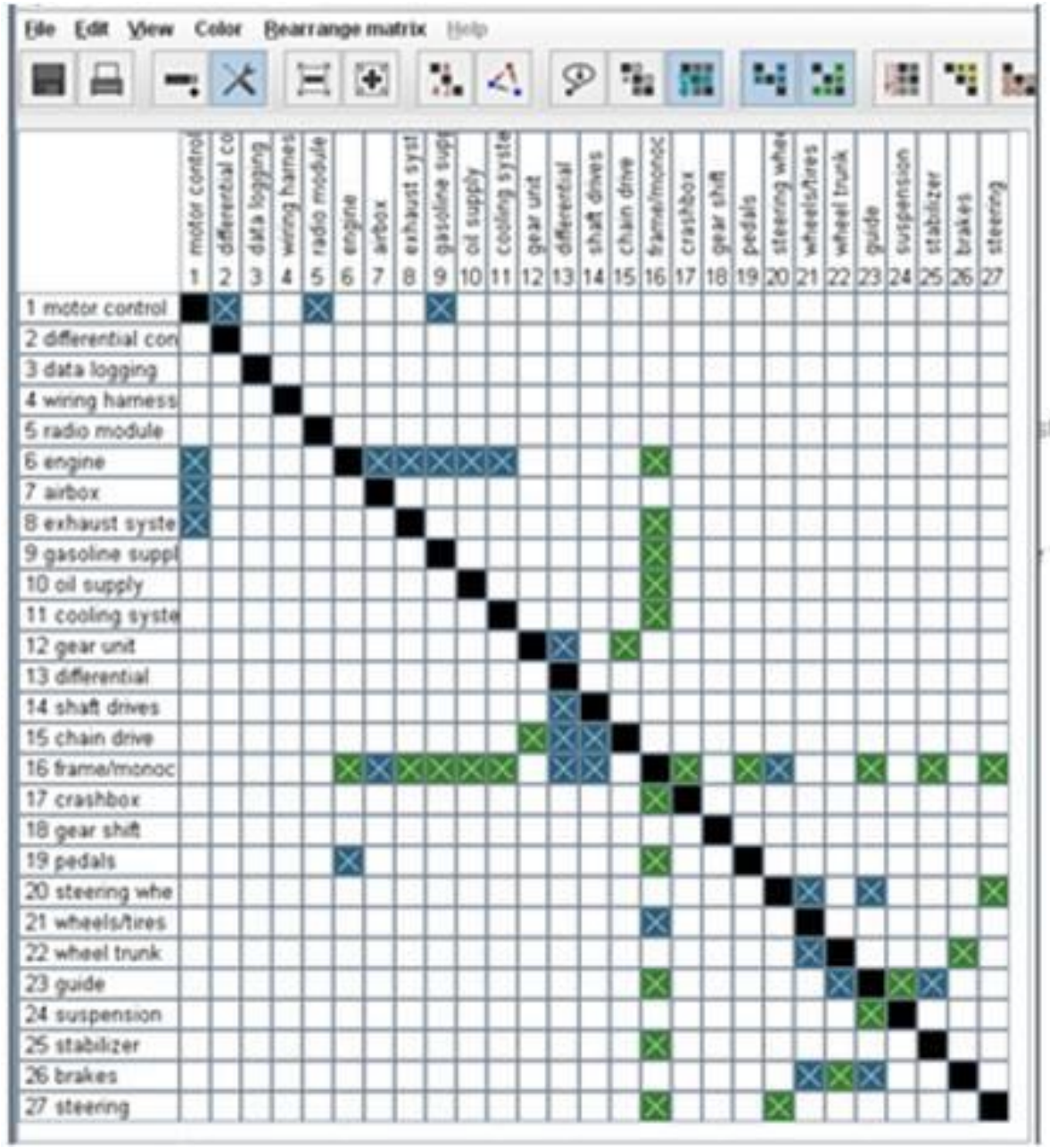


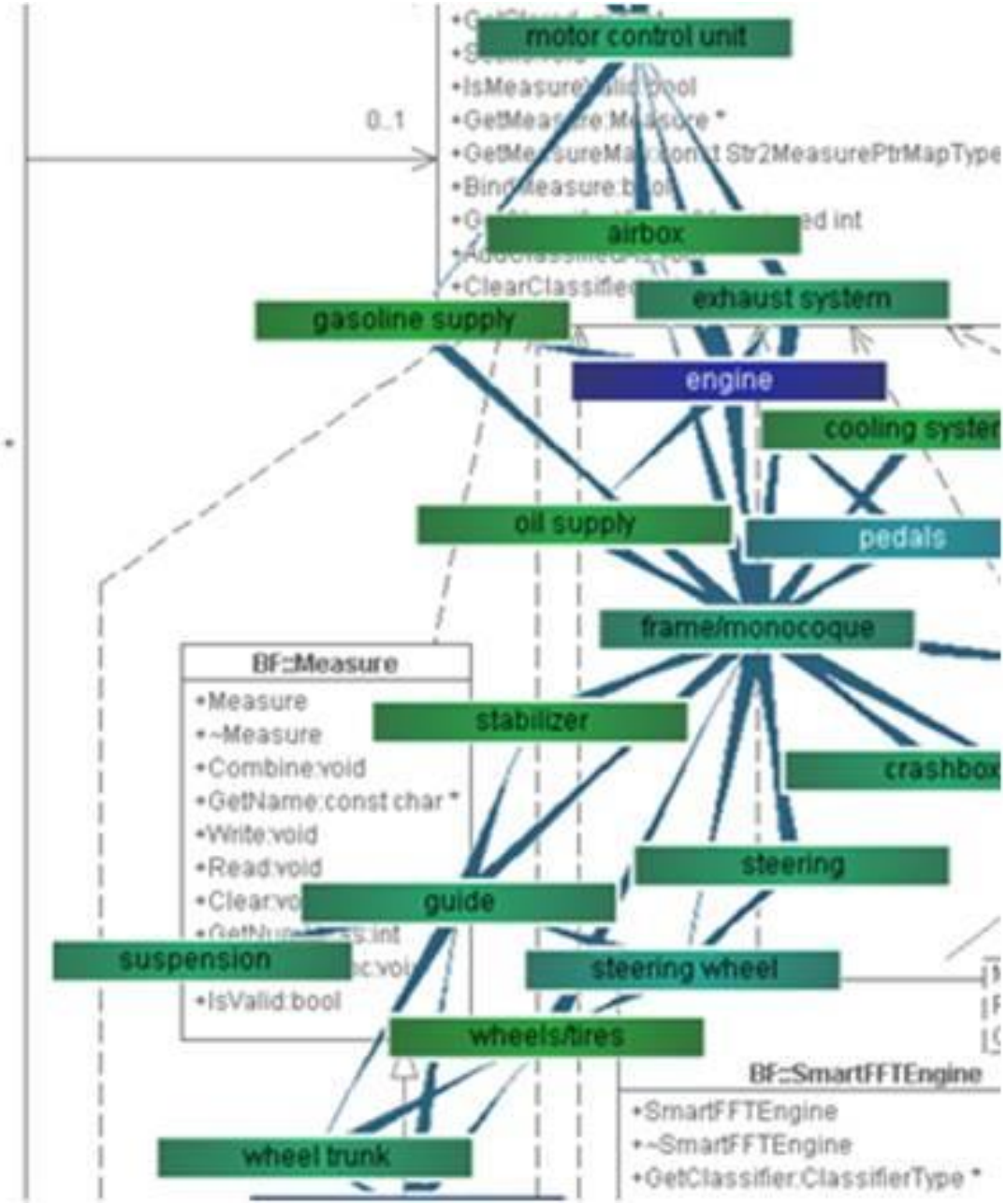
Fig. 4. Clustered team-based DSM for the ICE design process. Each mark indicates that the discipline in the row requires information from the discipline in the column due to one or more of the critical design trades (the loop types shown in Table 2).

	bolt 1	bolt 2	bolt 3	flange	base plate	cylinder body	cylinder
bolt 1				X	X		
bolt 2				X	X		
bolt 3				X	X		
flange					X	X	
base plate							
cylinder body							X
cylinder							



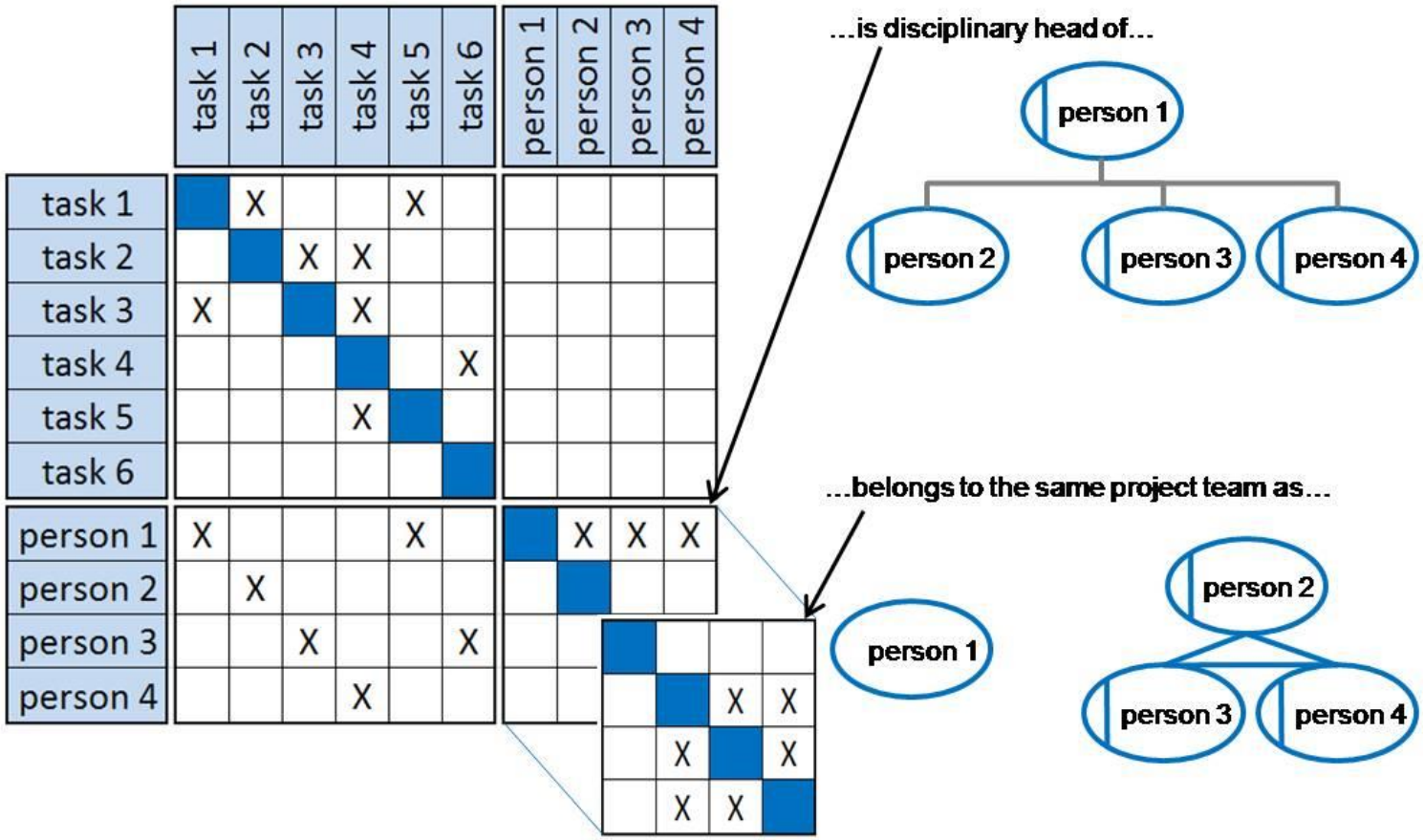
https://dsmweb.org





https://dsmweb.org

Multi-dimension DSM



- MDM allows analyzing a system's structure across multiple domains, condensing each single analysis into one DSM that represents multiple domains at a time.

Other flavors

House of Quality
FlyCo Drones

	Customer Specifications / Importance		Styling Jury	Weight	Size	Noise Output (dB)	Horizontal Pixels	Gyroscope Variation	Kilometers	W26 Chip	W11 Chip	Infrared Sensors	Stress Tests	Minutes	Comparitive Assessment 1	2	3	4	5
	Maximize or Minimize		▲	▼	▼	▼	▲	▼	▲	▲	▲	▲	▲	▲					
	Engineering Specfications		5-8 People	1.81 kg	33 x 33 x 15.24 cm	27 dBA	4K	-3 deg	6.27 Km	2 Chips	1 Chip	21 Sensors	29 kg/sq m	26 min					
1	Sylish	2	◎	△													F	P	S
2	Lightweight	3		◎	◎	△								△		P		S	F
3	Quiet	5				◎									P			S	F
4	Super HD Camera	5					◎							❍			P	F	S
5	Stable	5						◎						❍			S	F	P
6	Long Range	4							◎					❍			P	F	S
7	Self-Directing	4								◎	❍			◎			F	S	P
8	Programmable Paths	5								△	◎	◎		❍			S	F	P
9	High-Quality Finish	2	❍	❍													F	P	S
10	Durable Body	3	❍			◎							◎			F	S		P
11	Long Drone Battery Life	5							△	❍	❍			◎	S		P	F	
12	Long Remote Battery Life	5								△				◎		S		P	F
	Importance		23	35	27	57	45	45	51	71	54	45	27	154	634				
	Importance Weight (%)		4%	6%	4%	9%	7%	7%	8%	11%	9%	7%	4%	24%	100%				

Legend

- ◎ Strong Relationship (9)
- ❍ Medium Relationship (3)
- △ Weak Relationship (1)
- ++ Strong Positive Correlation
- \+ Positive Correlation
- \- Negative Correlation
- -- Strong Negative Correlation
- F FlyCo
- S Samsung
- P Panasonic

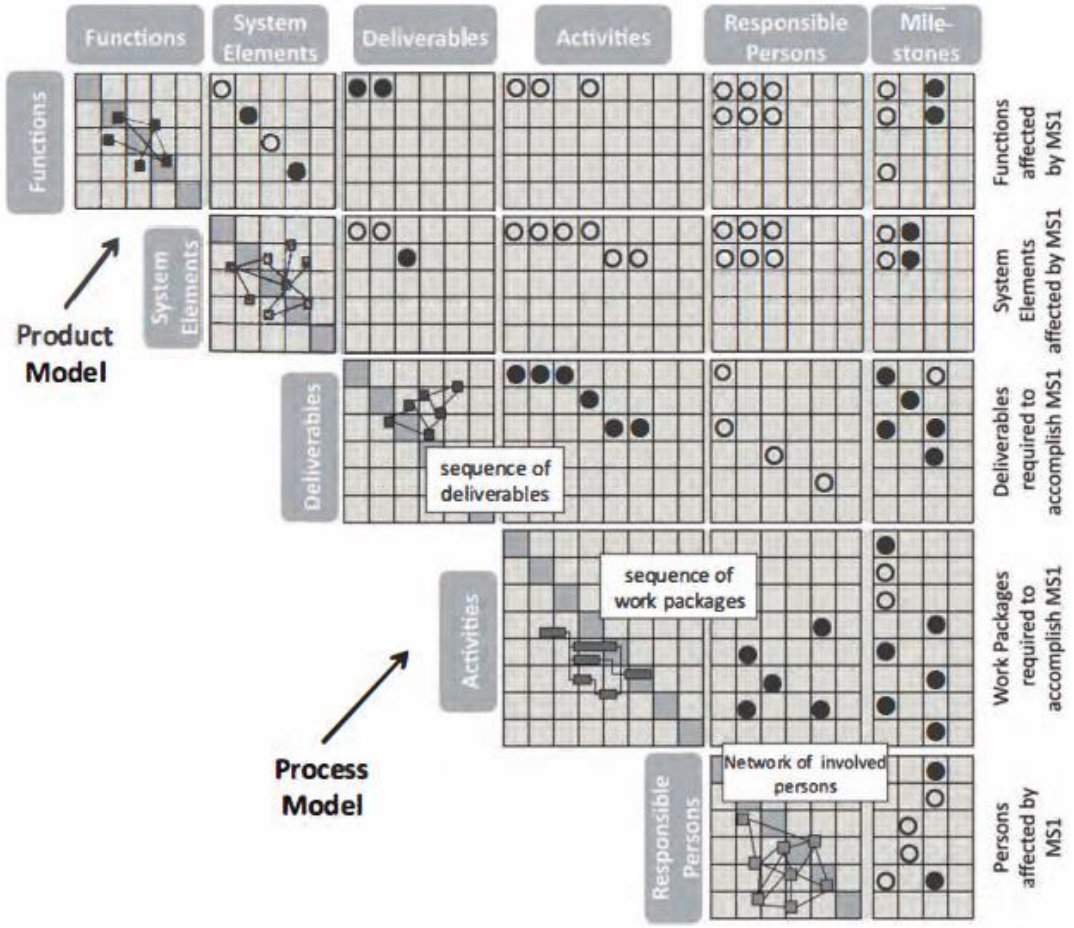
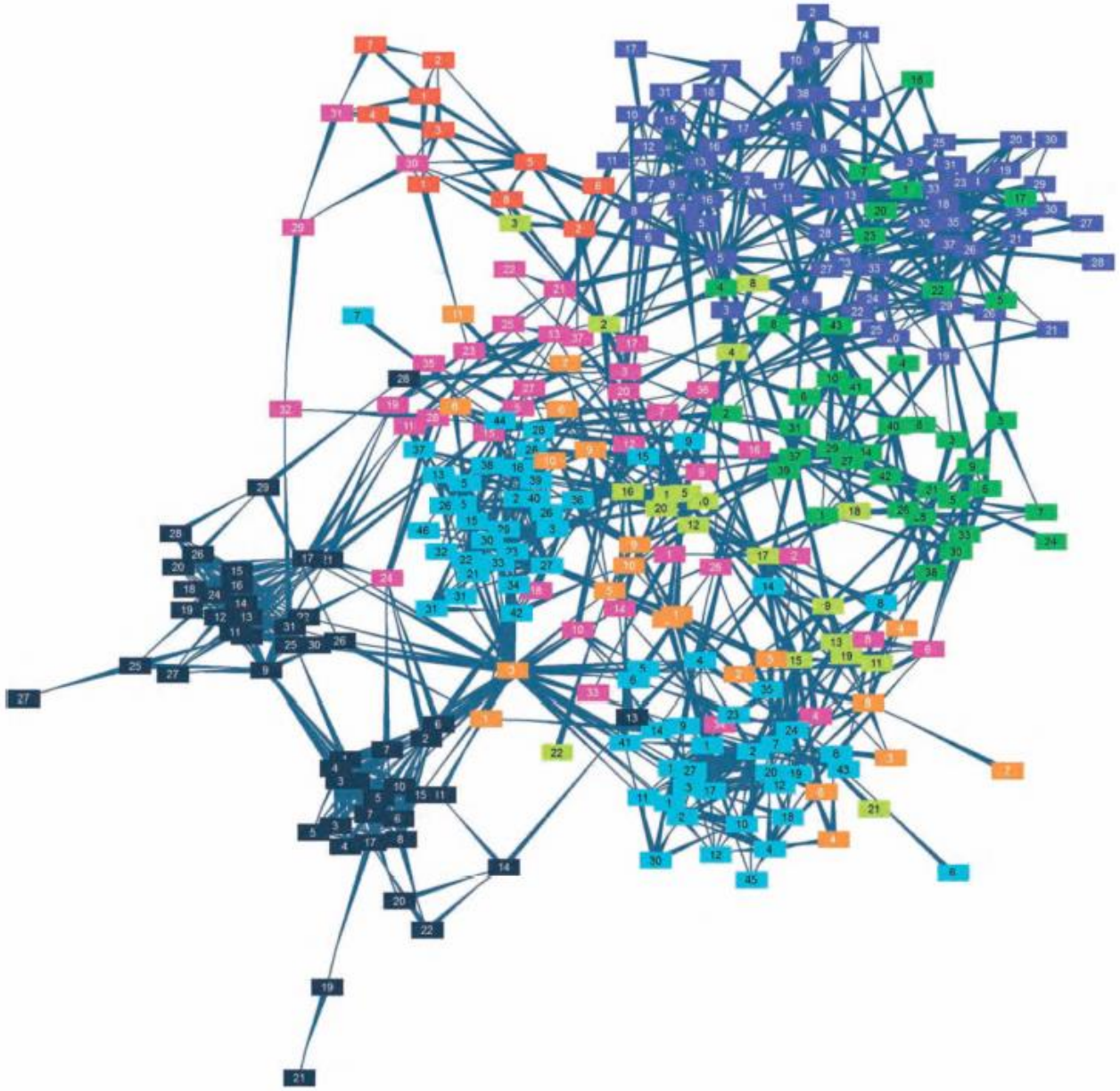


Figure 9.6.2
Layout of the MDM used to analyze the product and its development process.

On software

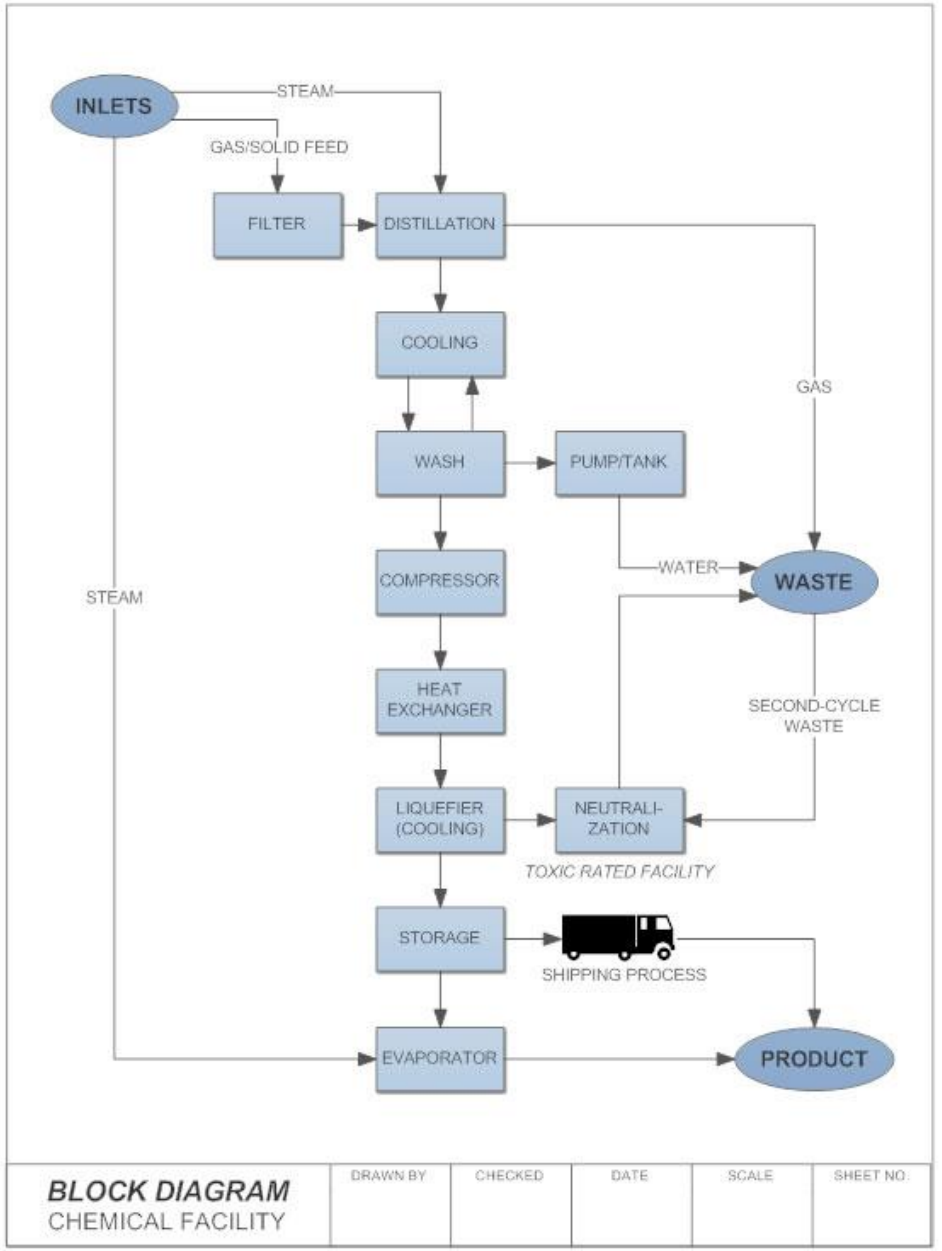
		Design Parameters							Relational Characteristics							
		Mixer	Connector	Dispenser	Foil Cartridge Holder	Foil Pack	Foil Cartridge	Mortar	Mixer	Connector	Dispenser	Foil Cartridge Holder	Foil Pack	Foil Cartridge	Mortar	System
Design Parameters	Mixer															
	Connector															
	Dispenser															
	Foil Cartridge Holder															
	Foil Pack												***			
	Foil Cartridge															
	Mortar															
Relational Characteristics	Mixer															
	Connector															
	Dispenser															
	Foil Cartridge Holder															
	Foil Pack															
	Foil Cartridge															
	Mortar															
	System															



Block Diagram

What is a block diagram?

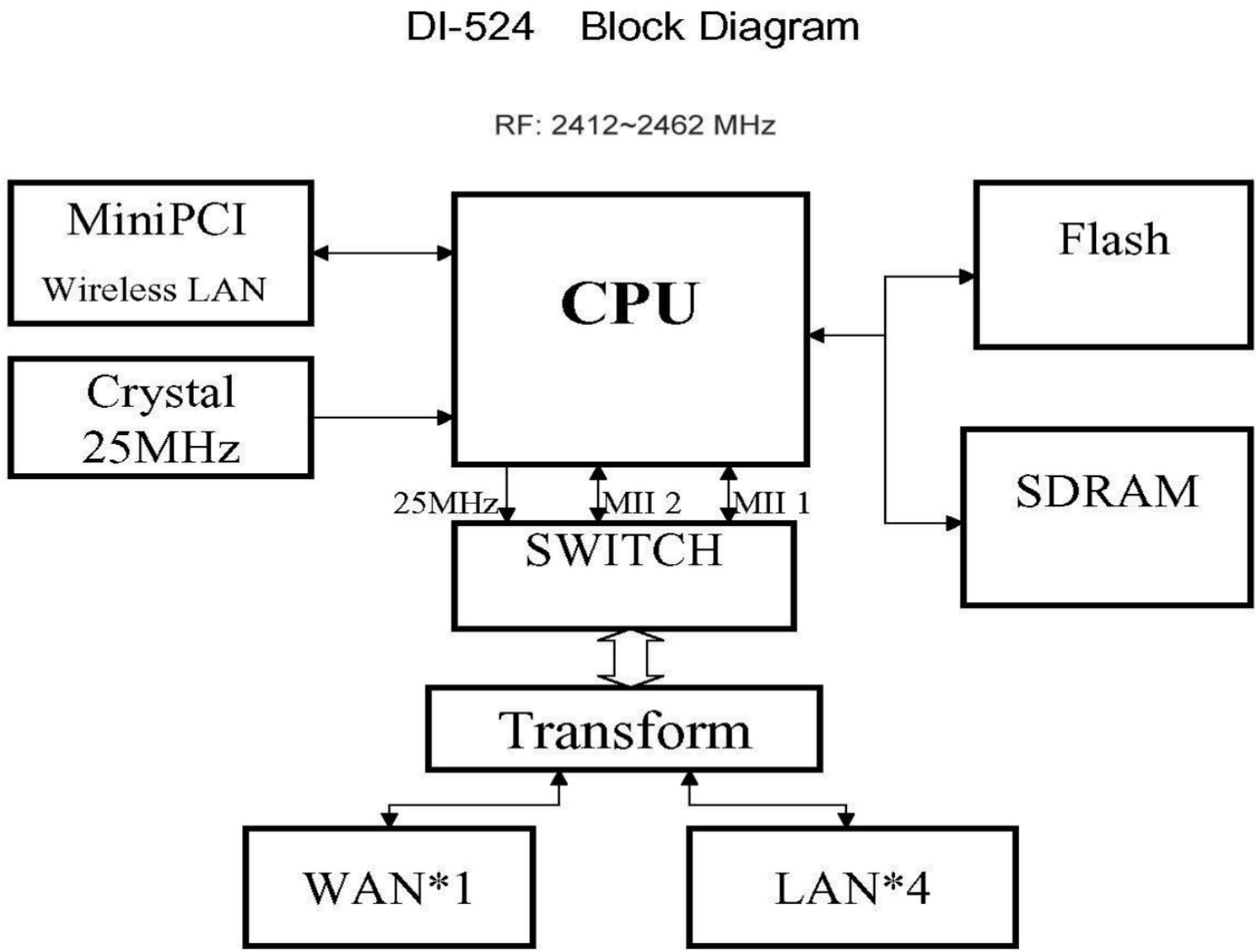
- A block diagram is a **specialized flowchart** used in engineering to visualize a **system at a high level**.
- Create your block diagram to **identify the most important components of your system** so you can focus on rapidly pointing out potential trouble spots.
- A block diagram is especially useful for **visualizing the inputs and outputs of your system**, while what happens in-between can remain in a black box.



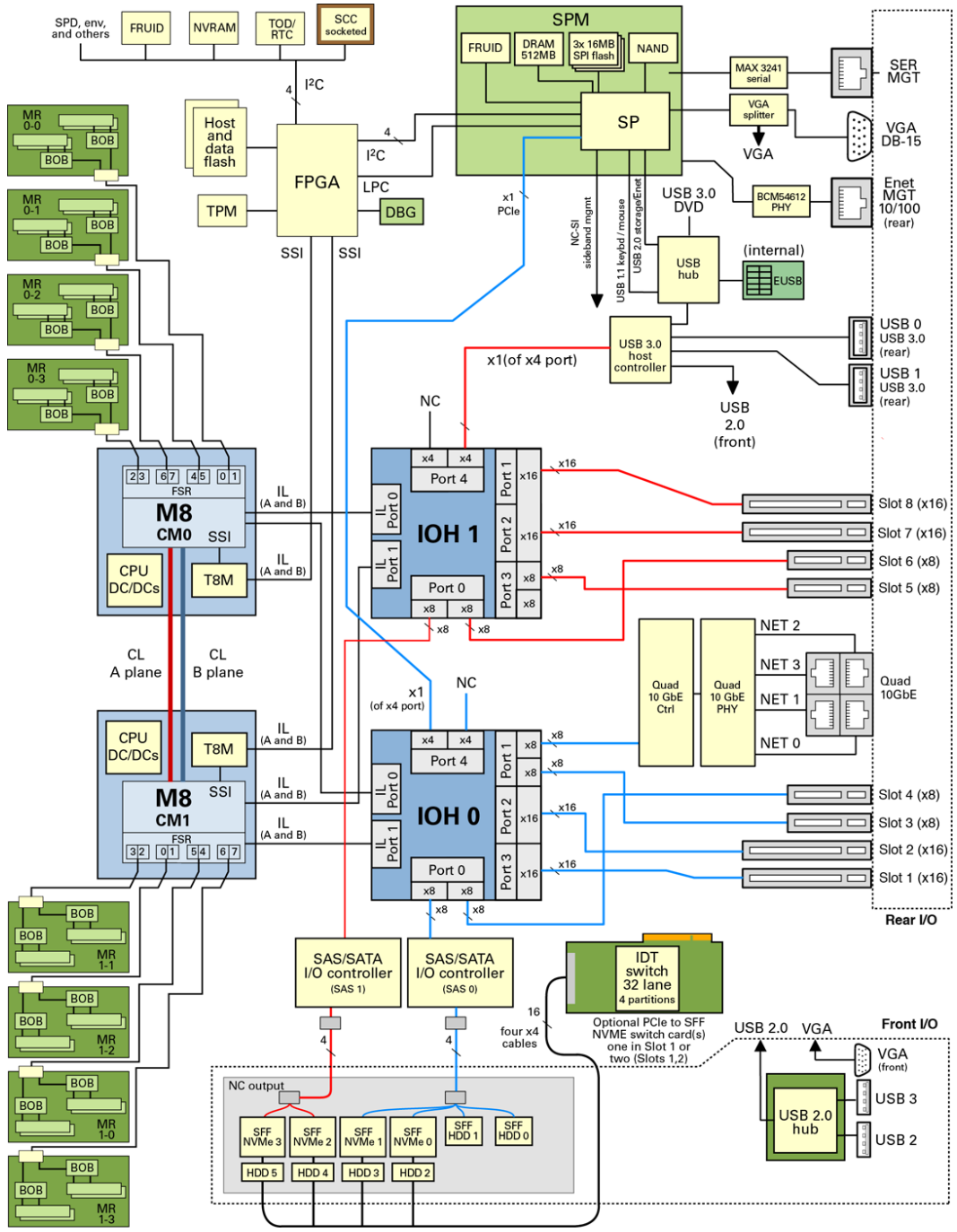
Symbols

- Block diagrams use very basic geometric shapes: **boxes and circles**. The principal parts and functions are represented by blocks connected by **straight and segmented lines** illustrating relationships.
- When block diagrams are used in electrical engineering, the arrows connecting components represent the direction of signal flow through the system.
- Whatever any specific block represents should be written on the inside of that block.
- *A block diagram can also be drawn in increasing detail if analysis requires it.* Feel free to add as little or as much detail as you want using more specific symbols.

Example:


DI-524 Block Diagram
RF: 2412~2462 MHz
MiniPCI
Wireless LAN
Crystal
25MHz
CPU
Flash
SDRAM
25MHz
MII 2
MII 1
SWITCH
Transform
WAN*1
LAN*4

Example:


SPD, env, and others
FRUID
NVRAM
TOD/ RTC
SCC socketed
SPM
FRUID
DRAM 512MB
3x 16MB SPI flash
NAND
SP
MAX 3241 serial
VGA splitter
VGA
SER MGT
VGA DB-15
Enet MGT 10/100 (rear)
BCM54612 PHY
USB 3.0 DVD
USB hub
(internal)
EUSB
USB 1.1 keybd / mouse
USB 2.0 storage/Enet
NC-SI sideband mgmt
x1 PCIe
4 I²C
Host and data flash
FPGA
TPM
DBG
LPC
I²C
SSI
SSI
MR 0-0
MR 0-1
MR 0-2
MR 0-3
BOB
USB 3.0 host controller
x1(of x4 port)
USB 0 USB 3.0 (rear)
USB 1 USB 3.0 (rear)
USB 2.0 (front)
NC
M8 CM0
SSI
FSR
CPU DC/DCs
T8M
IL (A and B)
IL Port 0
IL Port 1
IOH 1
Port 4
Port 0
Port 1
Port 2
Port 3
x4
x8
x16
Slot 8 (x16)
Slot 7 (x16)
Slot 6 (x8)
Slot 5 (x8)
CL A plane
CL B plane
x1 (of x4 port)
NC
Quad 10 GbE Ctrl
Quad 10 GbE PHY
NET 2
NET 3
NET 1
NET 0
Quad 10GbE
M8 CM1
IOH 0
Slot 4 (x8)
Slot 3 (x8)
Slot 2 (x16)
Slot 1 (x16)
Rear I/O
SAS/SATA I/O controller (SAS 1)
SAS/SATA I/O controller (SAS 0)
IDT switch 32 lane 4 partitions
Optional PCIe to SFF NVME switch card(s) one in Slot 1 or two (Slots 1,2)
16 four x4 cables
NC output
SFF NVMe 3
SFF NVMe 2
SFF NVMe 1
SFF NVMe 0
SFF HDD 1
SFF HDD 0
HDD 5
HDD 4
HDD 3
HDD 2
USB 2.0
VGA
Front I/O
VGA (front)
USB 3
USB 2
USB 2.0 hub

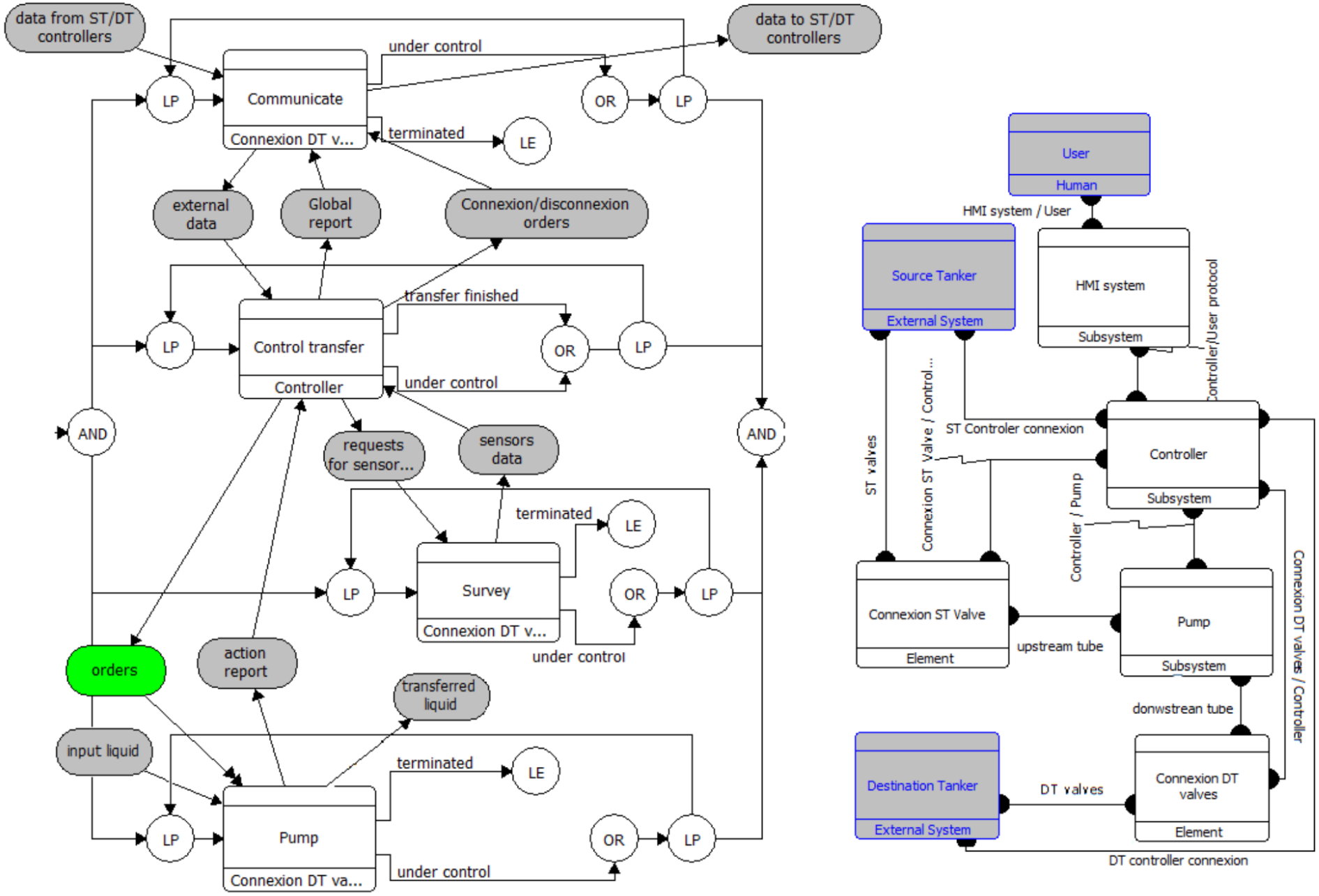


Figure 9. A functional (left) and physical (right) architecture of a liquid transfer system.

https://www.researchgate.net/publication/308088265_Towards_VV_suitable_Domain_Specific_Modeling_Languages_for_MBSE_A_tooled_approach

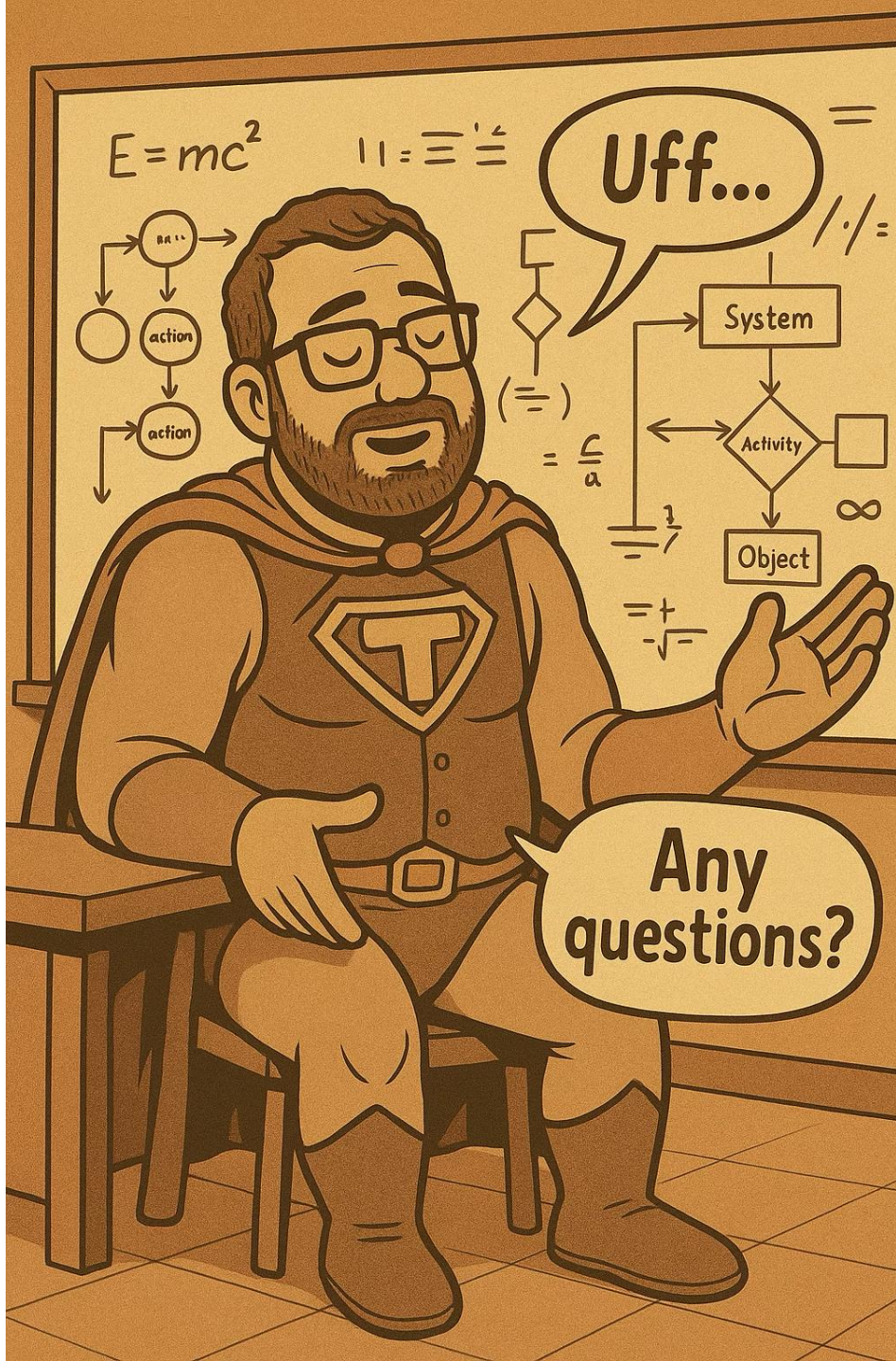
Final Considerations

Last words - Exercises

- Self-paced (no due-date – just joy)
  - A list of 20 cases to apply the classical diagrams.
- To deliver
  - A list of 10 conceptual questions at the Classroom (each click counts --- click Only if you are sure of the answer)
- To deliver (class leader)
  - Send me a list of the groups w/ selected themes (asap)
  - Place on the collaborative document at the GClass, the names and links for the presentation / report.

Last words

- Hope you enjoy these lectures
- Keep in mind that (Model based) Systems Engineering is one of EMBRAER’s verticals.
- I’m available through the GClass, and through the Class Leader.
- We will dive into the MBSE – it will be useful on the third phase.



Prof. Dr. Christopher Shneider Cerqueira
Model based Systems Engineering // Multidomain Research

starting...

"There is something" (MFS)

... on the journey to uncover the essence of abstraction.

Fast Links:

Example: Mosquito killing racket