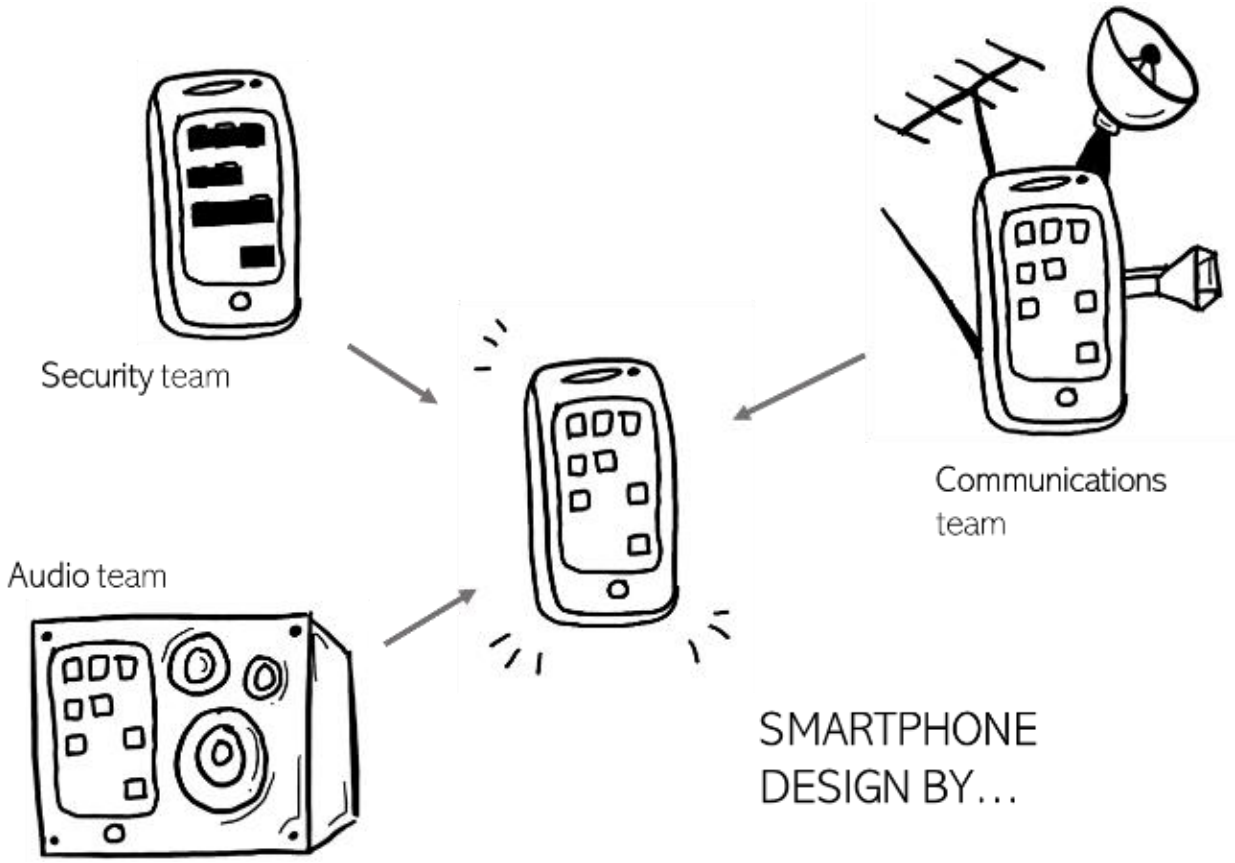
SYSTEMS ENGINEERING

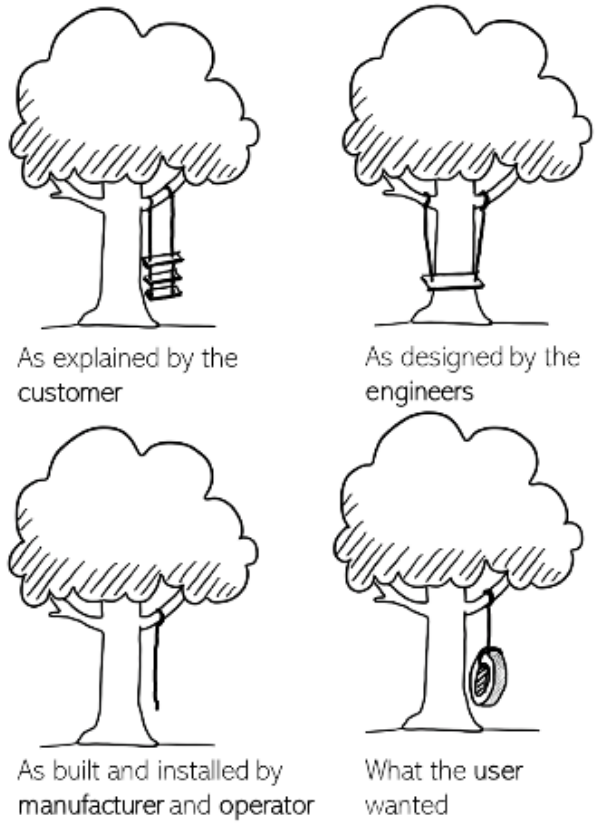
KISS One-Shot Missions Lecture

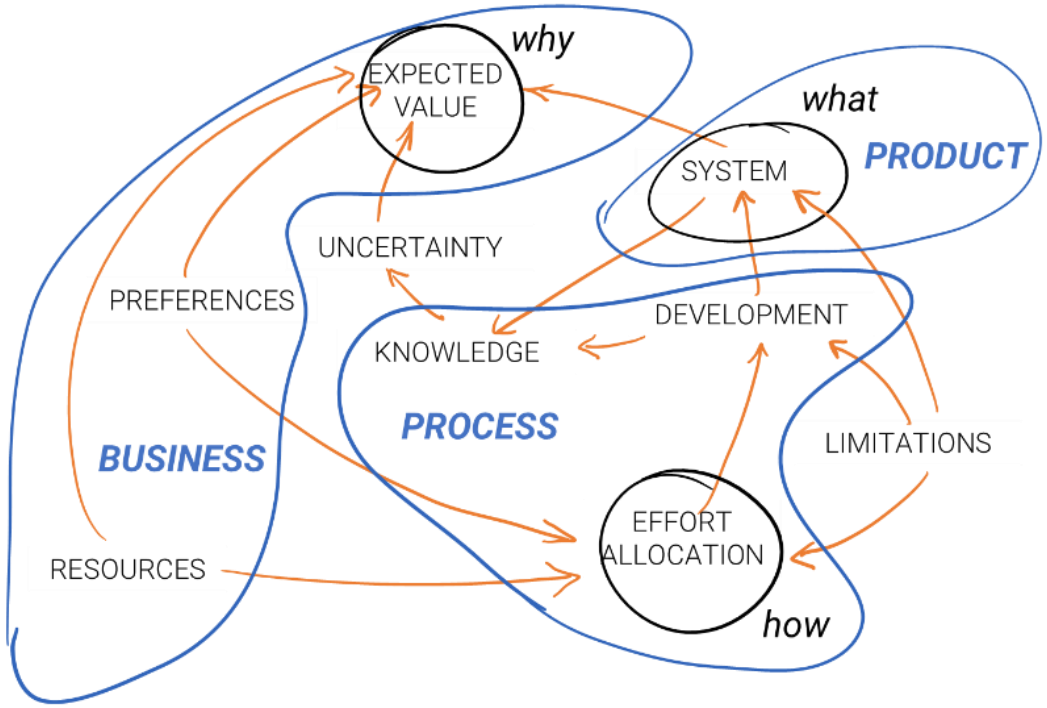
Dr. **Alejandro Salado**

Department of Systems and Industrial Engineering

THE UNIVERSITY OF ARIZONA







Think about the end before the beginning

Avoid unintended consequences

Anticipate, prevent, multi-path

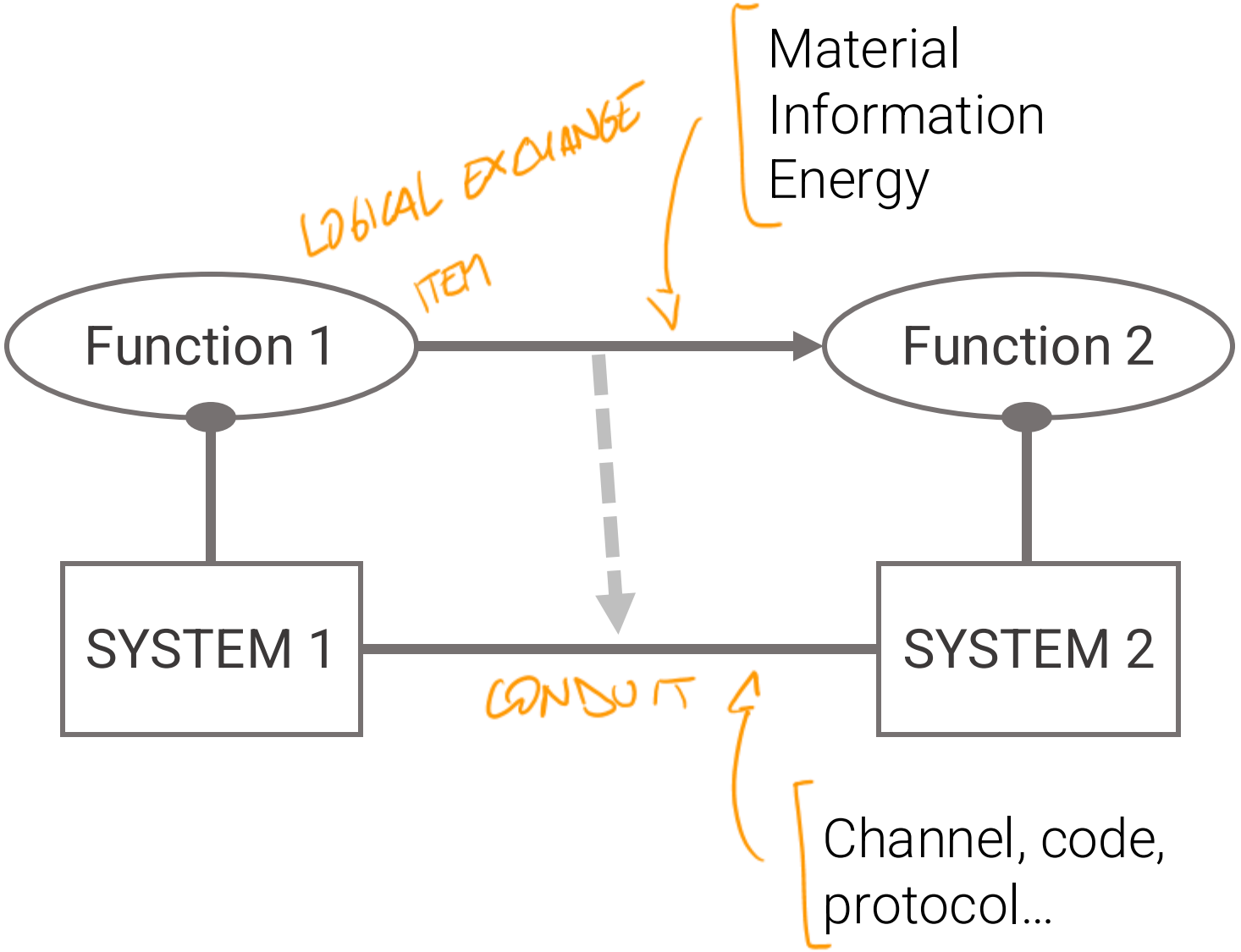
Stealth, persuade, influence, command

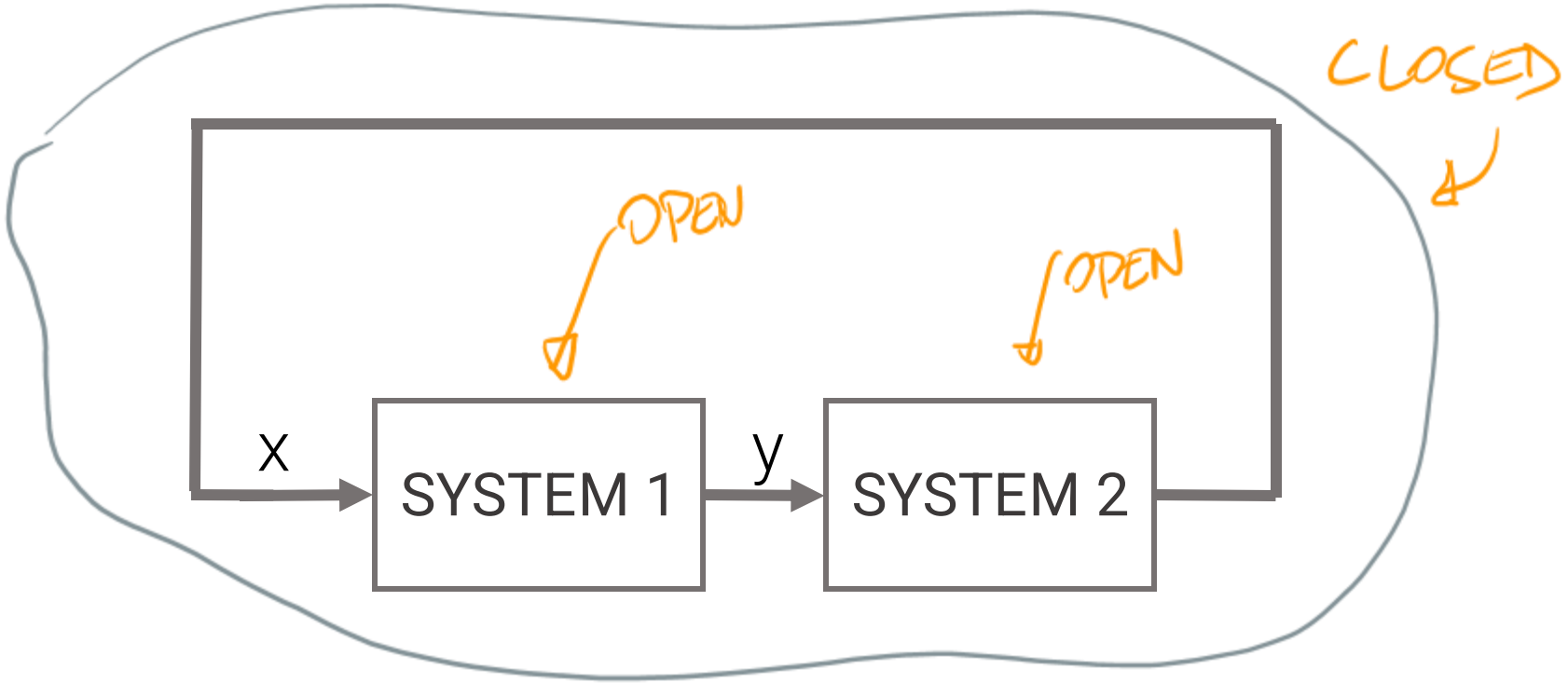
SE IS ALL ABOUT DESIGNING & EXECUTING STRATEGIES

Identify the problem

Strategize (architect) the solution

Demonstrate the effectiveness of the solution





FUNCTION → CAPABILITY → ENGAGEMENT → OUTCOME

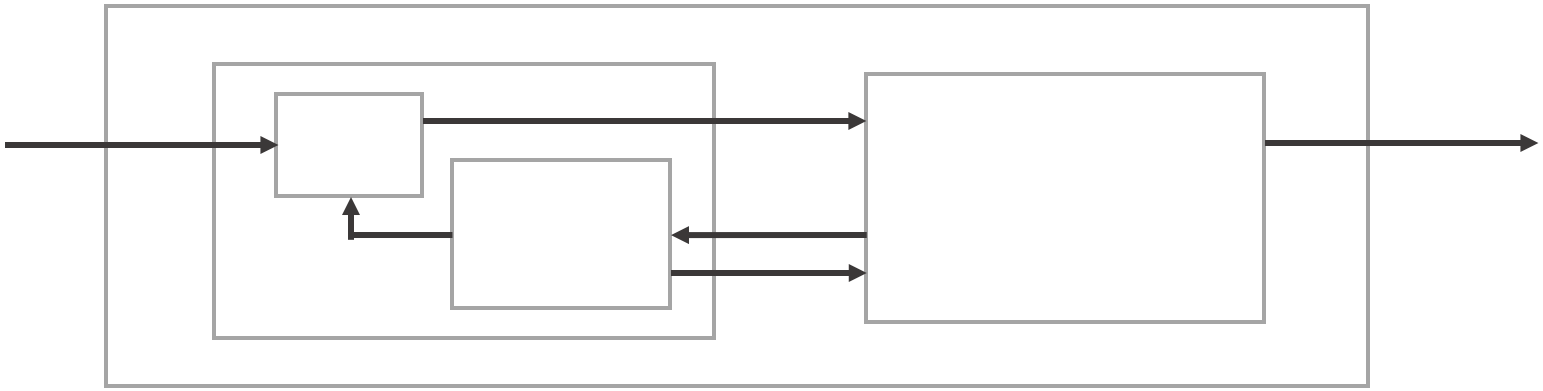
OPEN SYSTEM	Function, outputs
CLOSED SYSTEM	State, outcomes

VALUABLE FOR SOMEONE

ENCAPSULATION

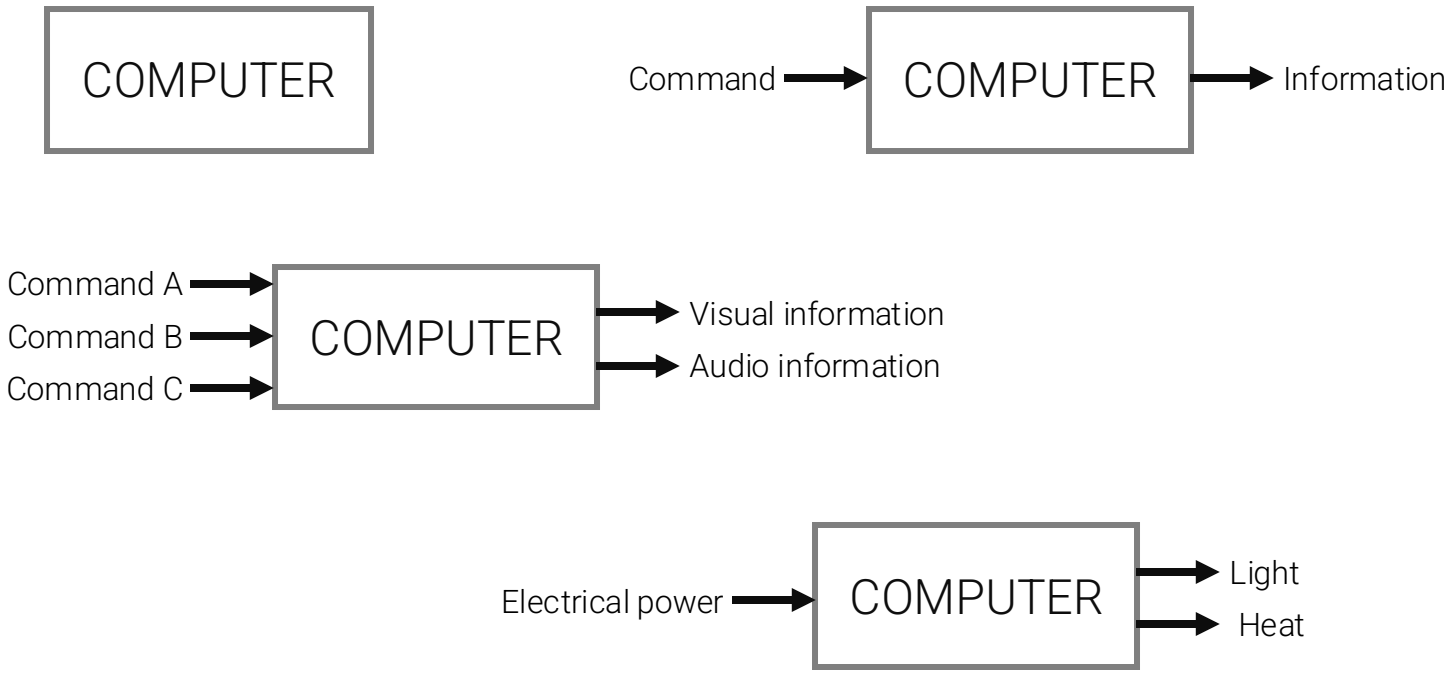
- System
  - Subsystem
    - Assembly
      - Equipment
        - Component
          - Part

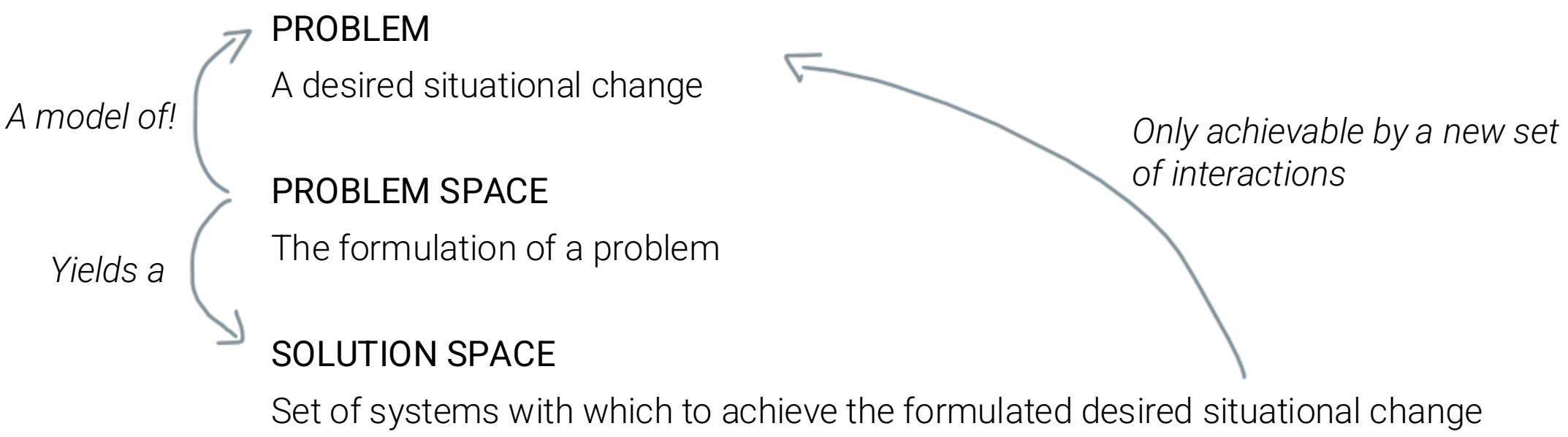
- System L0
  - System L1
    - System L2
      - System L3
        - System L4
          - System L5



ABSTRACTION

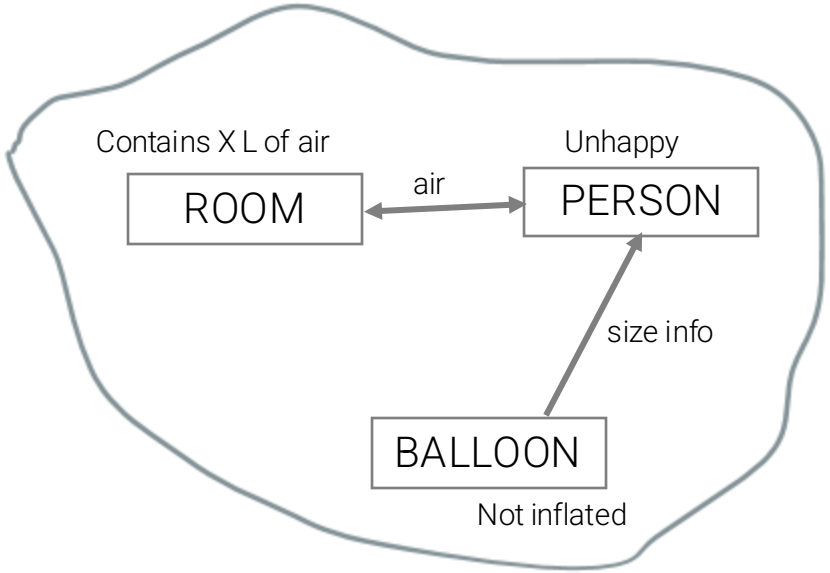
Level of detail **at a given level** of encapsulation



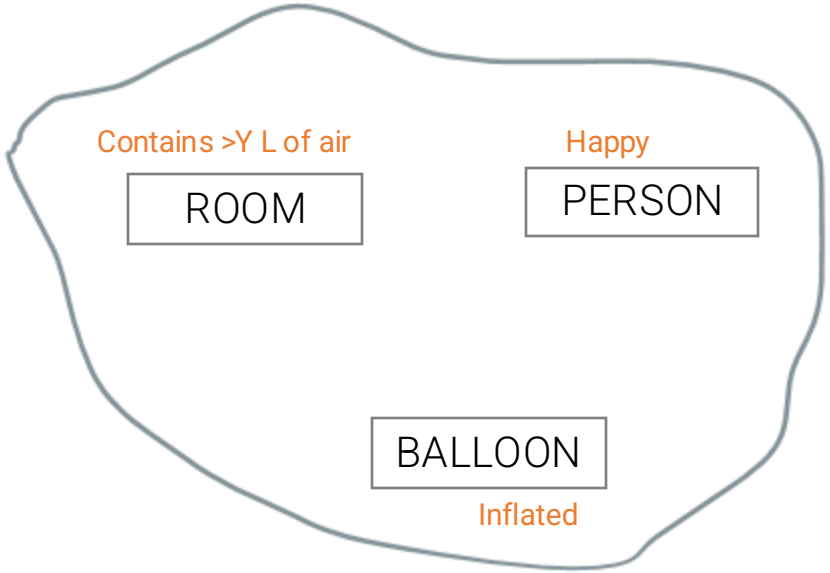


KINDS OF PROBLEMS

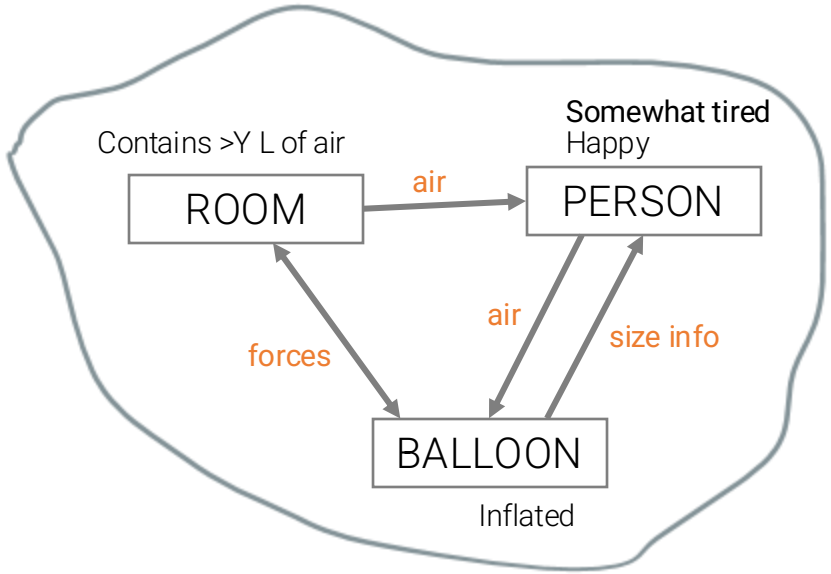
OUTCOMES	EXCHANGES
Be happier	Get a soda after pushing a button
Detect an intruder	Retrieve information after accessing a book
Complete activity in less than X h	Produce data that captures incoming light
Move from A to B	Eject ink after injecting pressure
Needs	*Requirements*



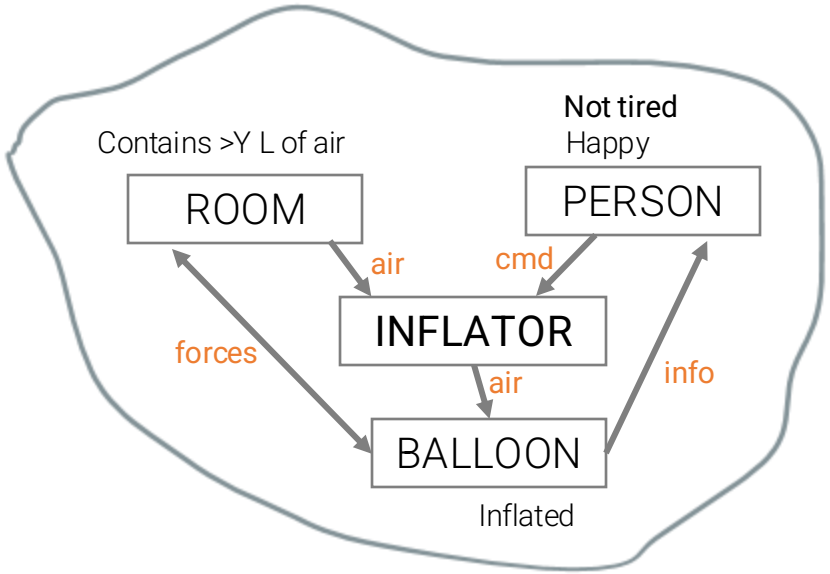
CURRENT situation



DESIRED situation



SOLUTION w/o additional system

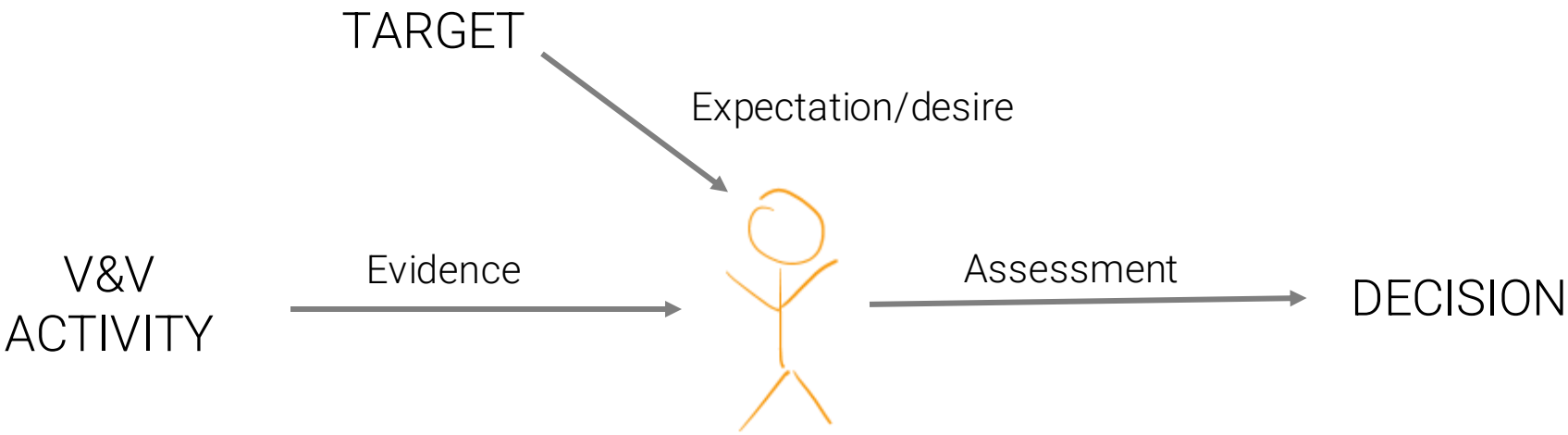


SOLUTION with additional system

Anything that provides **information** about **THE OUTCOMES BY THE SYSTEM** is VALIDATION

Anything that provides **information** about **THE SYSTEM** is VERIFICATION

HOW DO WE **USE INFORMATION**?



COGNITIVE process at the **center** of V&V

ARCHITECTURE

STRATEGY of the technical solution

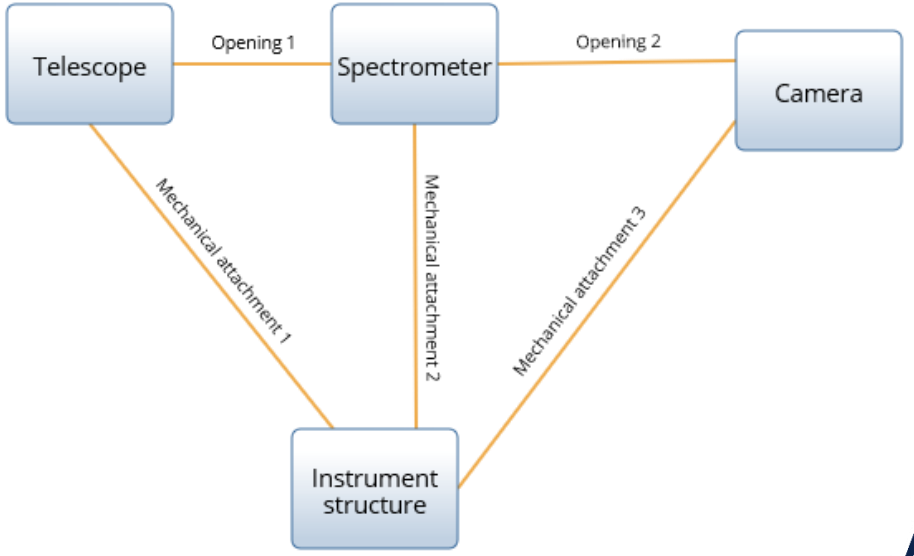
Drives **requirement** fulfillment
Drives **integrability** and **verifiability**.
Drives **contractual** requirements for subcontractors.
Drives **variant** design and management.
Drives **changeability**.
Drives ease of **understanding**.

MODEL

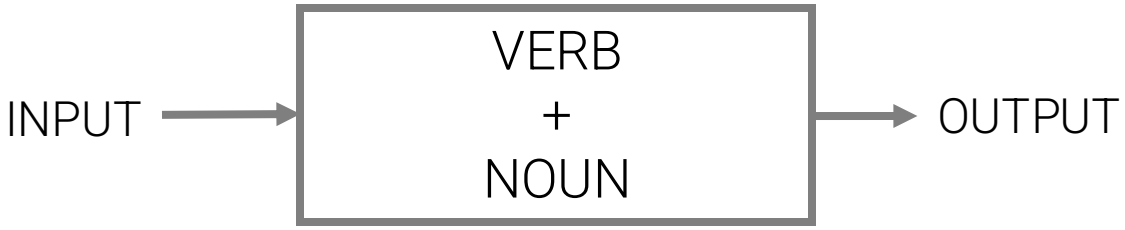
REPRESENTATION of aspects of reality

DIAGRAM

COMMUNICATION agreement



FUNCTIONAL VIEW



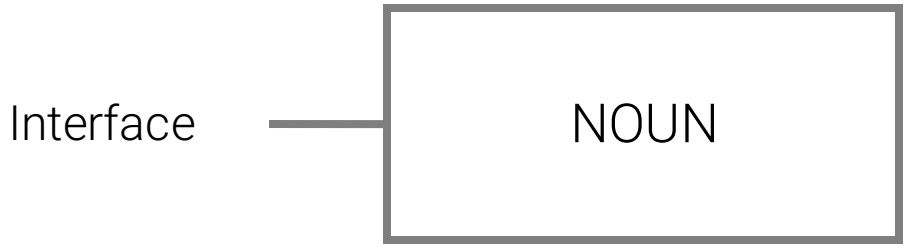
A function always describes an **ACTION**

A function **ALWAYS** has at least 1 input and 1 output

Inputs and outputs capture **LOGICAL**/conceptual items that the system/function processes

→ Inputs and outputs are FLOWS, not interfaces

PHYSICAL VIEW



A component describes a **MECHANISM** that executes an action

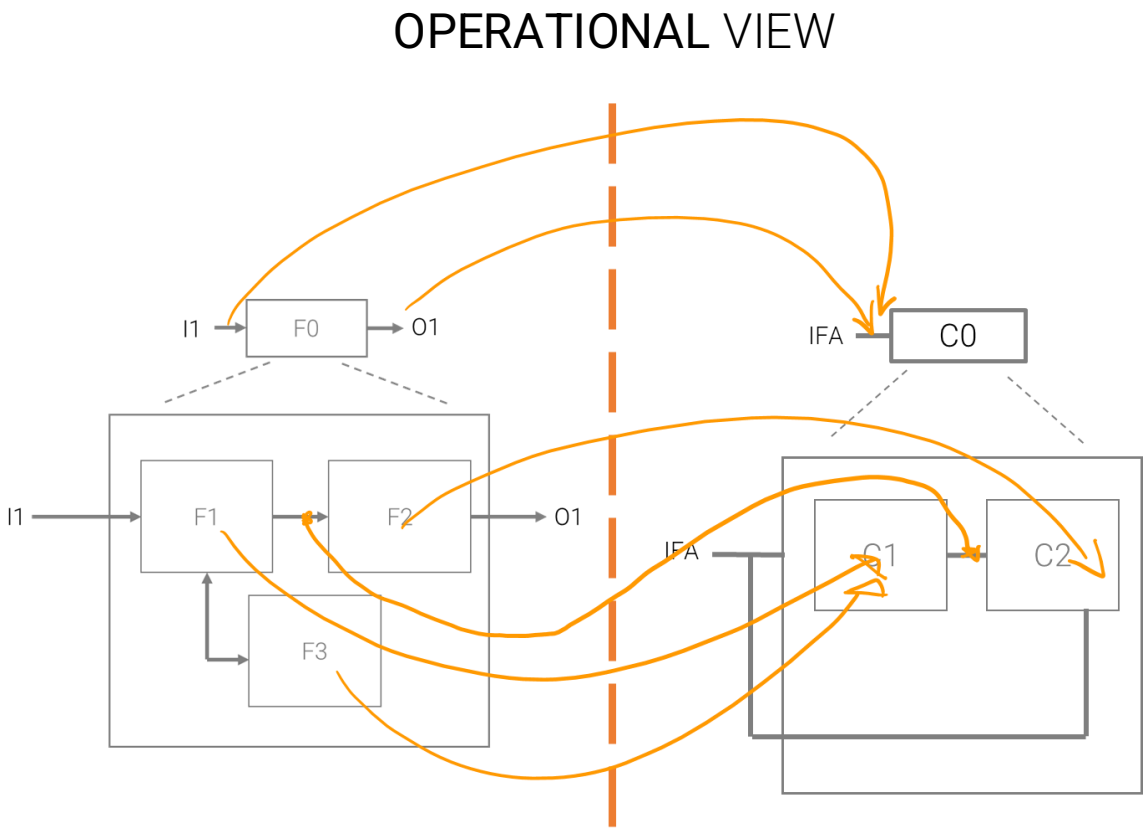
→ Examples: HW, **SW**, facilities, equipment, people…

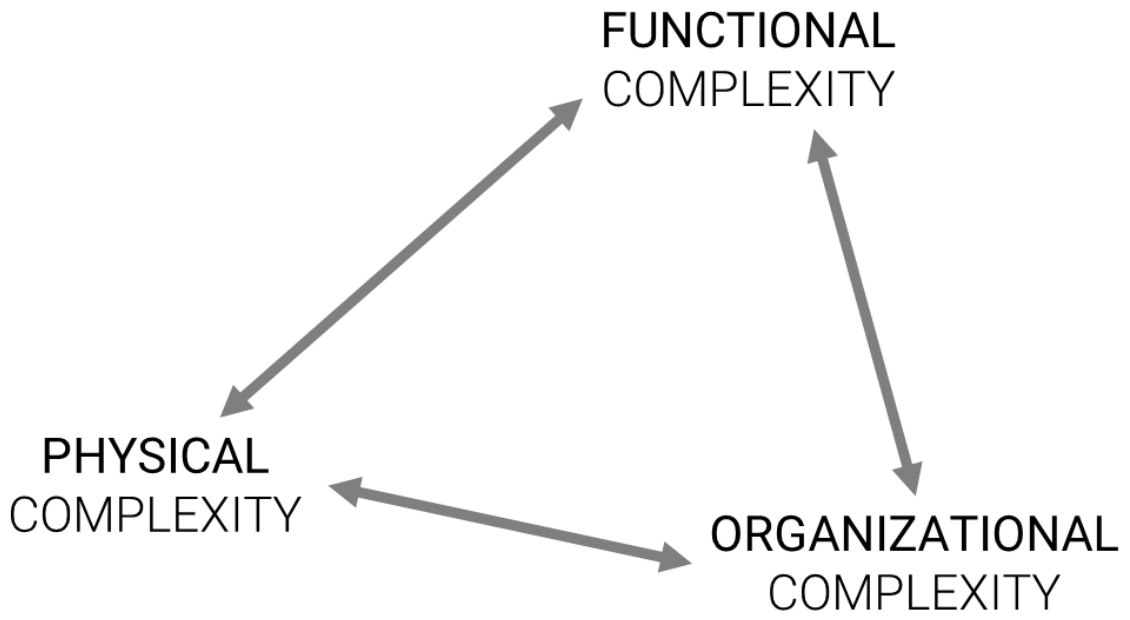
A component is in principle **FORMLESS**

A component **ALWAYS** has at least one interface

Interfaces capture the physical conduits that carry inputs and outputs

→ Interfaces are NON-DIRECTIONAL





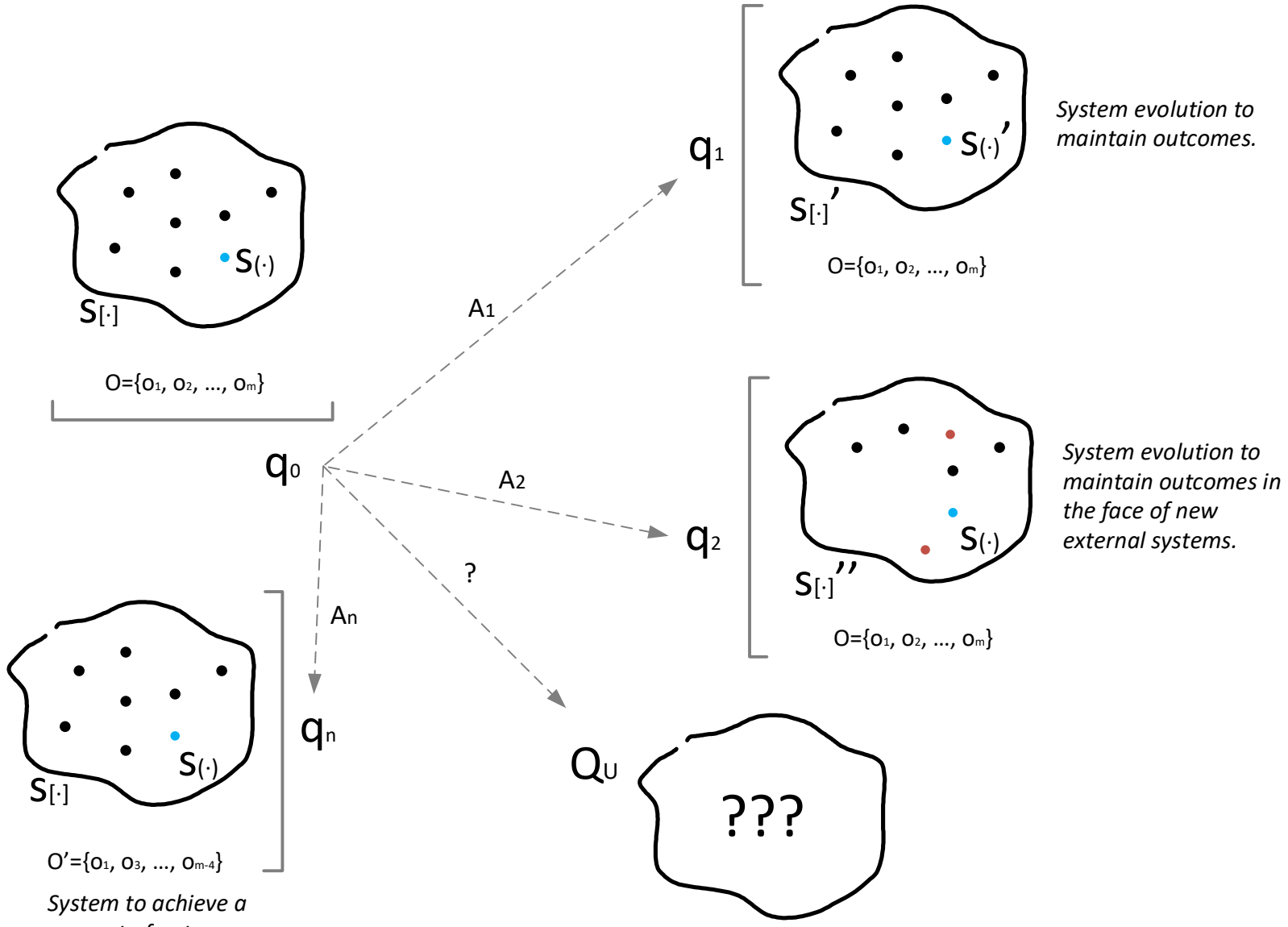
ARCHITECTING each dimension

&

ARCHITECTING the relations between the dimensions

What is an ility?

(Simplified)
Given a current situation, how likely the system is to reach desired situations in the future.



ONE-SHOT missions… so what?

FROM

I want to learn about the existence of vegetation in Mars.

So, I need to measure the IR radiation of Mars' surface.

TO

I might want to measure something about some celestial object at some point in time.

I will only tell you what I want after you have given me your solution!

1. How do we formally define this problem?
2. How do we architect a solution to this problem?
3. How do we know we will be able to measure something about something at some point?

Any **exemplar** to learn from?

1. PROBLEM
   Reproduction
2. ARCHITECTURE
   Simple core functionality, selective redundancy, extreme ilities (driven by resilience?), intelligence
3. V&V
   Evolution, trial and error (long-term and short-term)

SYSTEMS ENGINEERING must evolve to satisfactorily support the engineering of one-shot missions

1. Define extremely uncertain engineering problems in an actionable manner

2. Architect for extreme ilities

3. V&V potentials, not outcomes

THANK **YOU**

alejandrosalado@arizona.edu

THE UNIVERSITY OF ARIZONA

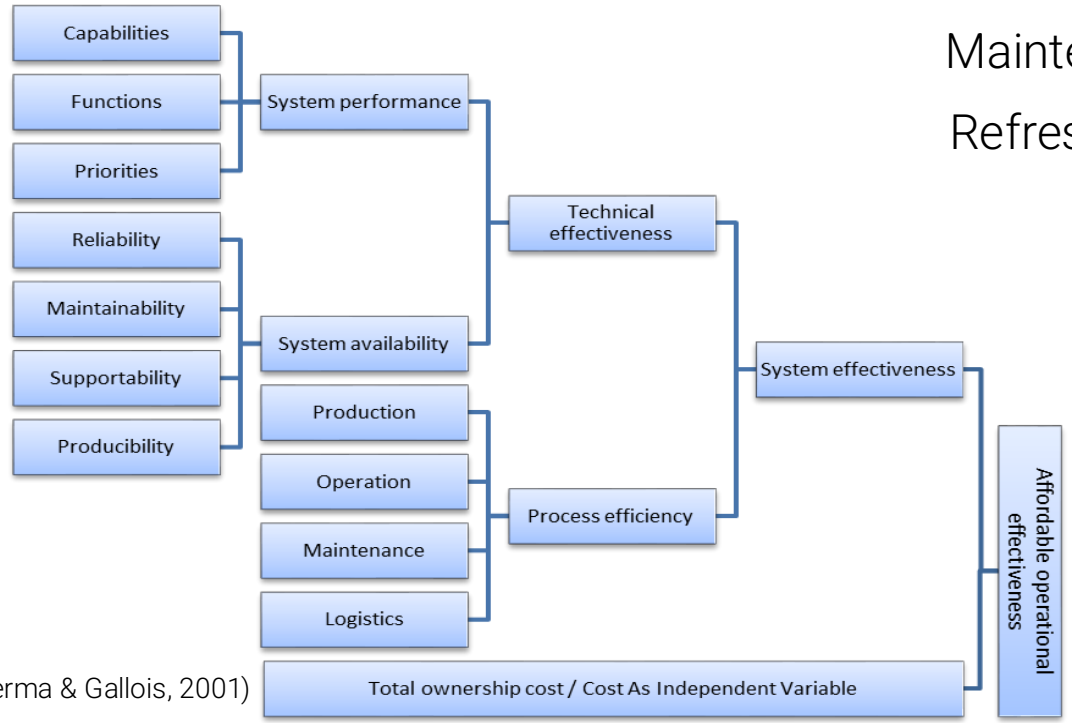
DEVELOPMENT
cost

OPERATIONAL
cost

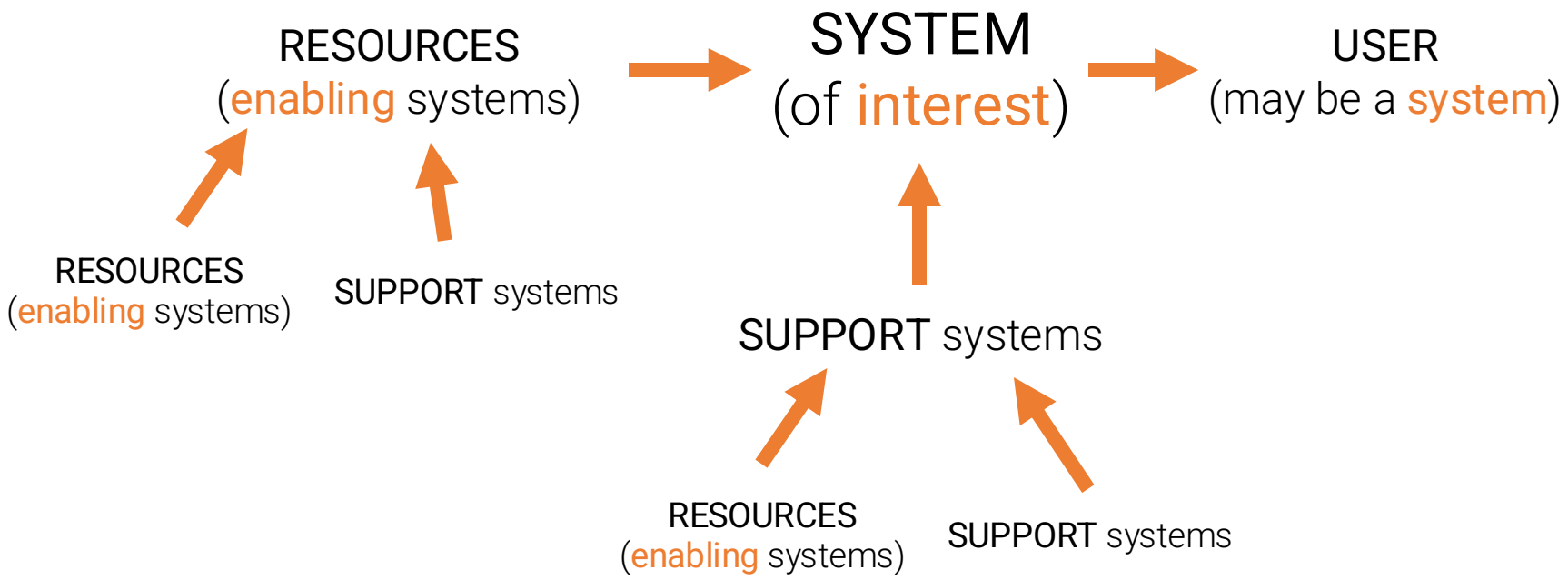
Use/Operation (nominal, non-nominal)

Maintenance (preventive/reactive)

Refresh (obsolescence/new needs)

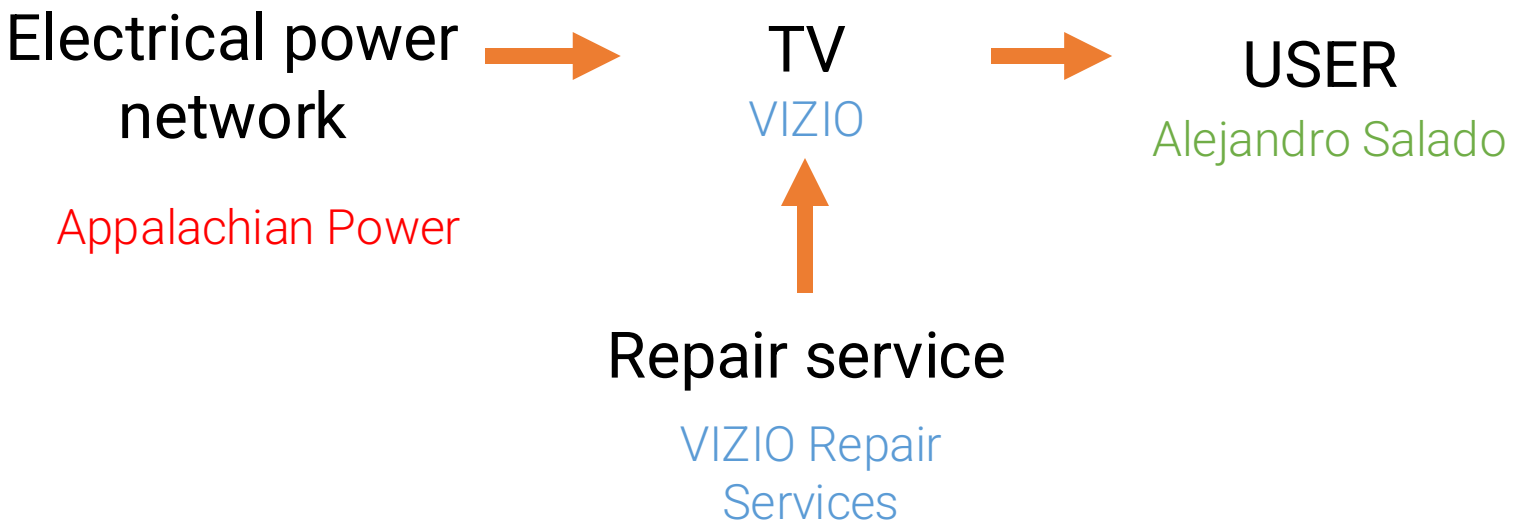


(Verma & Gallois, 2001)

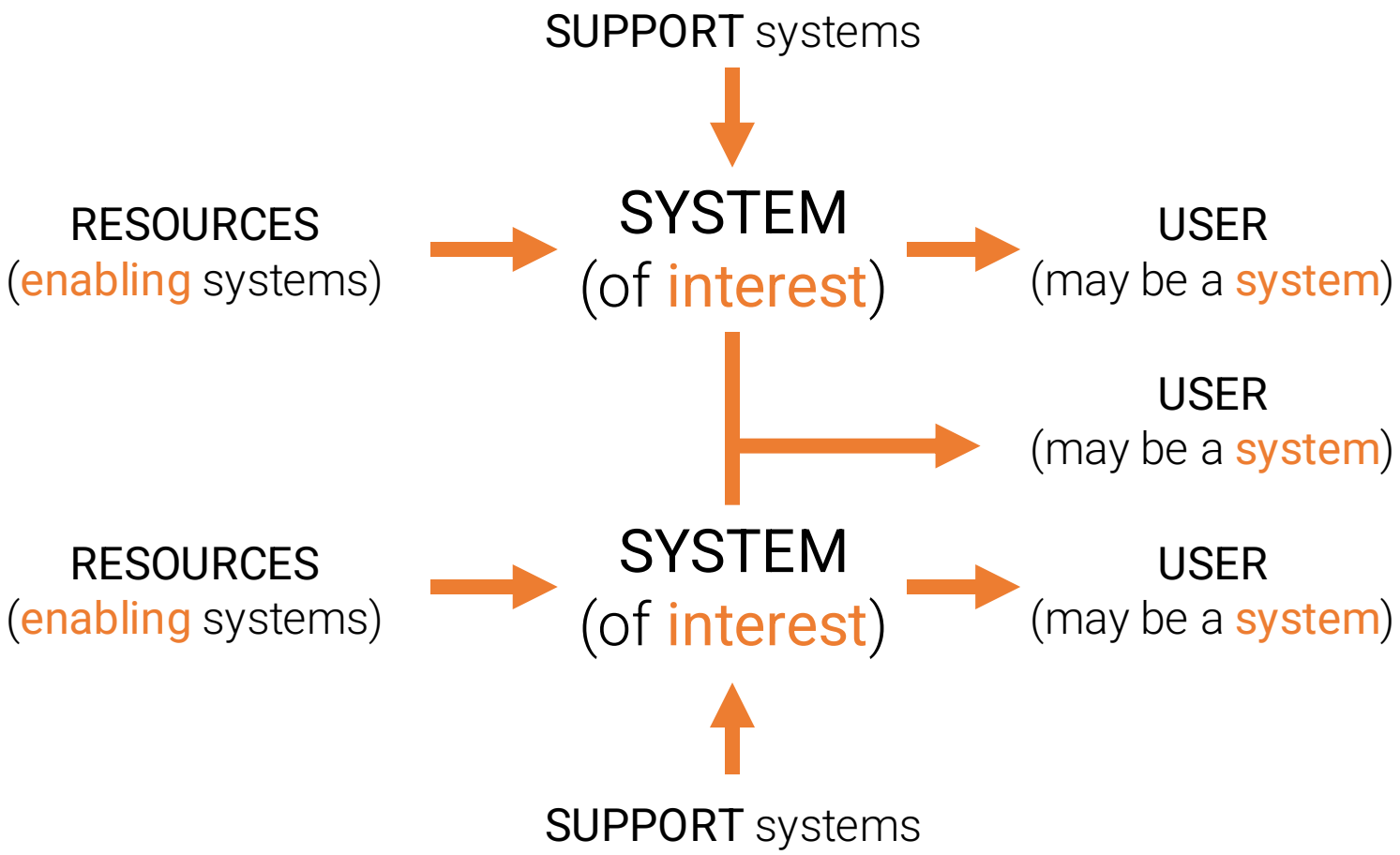


Is this a system of systems?

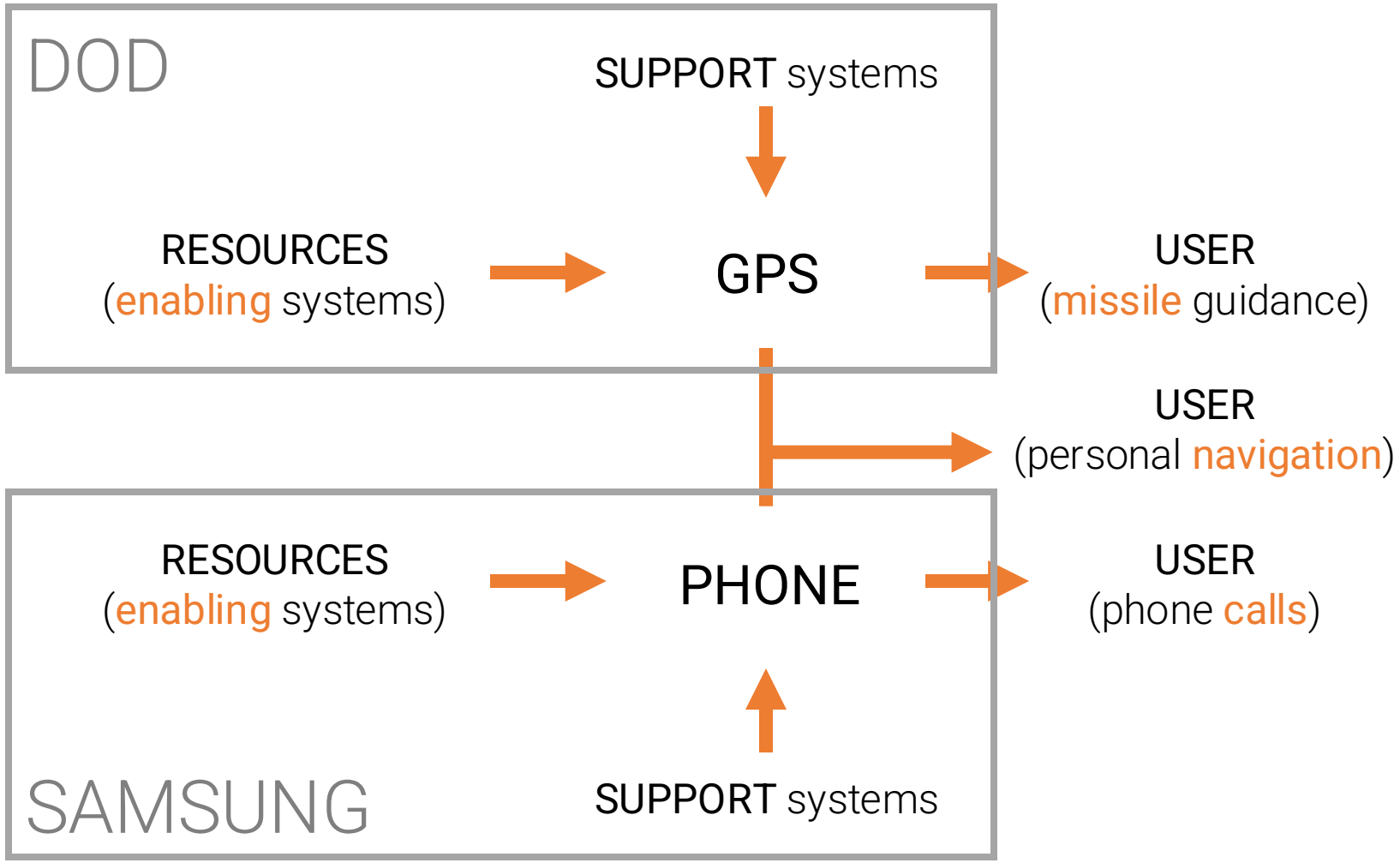
GOVERNANCE & PURPOSE



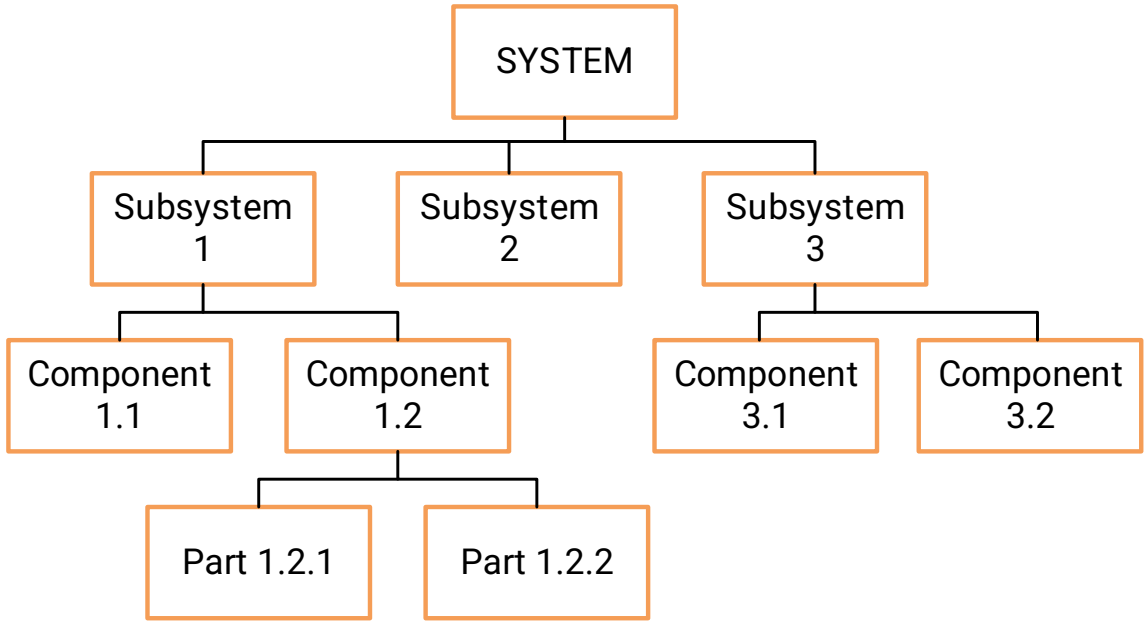
Is this a system of systems?



Is this a system of systems?



Is this a system of systems?



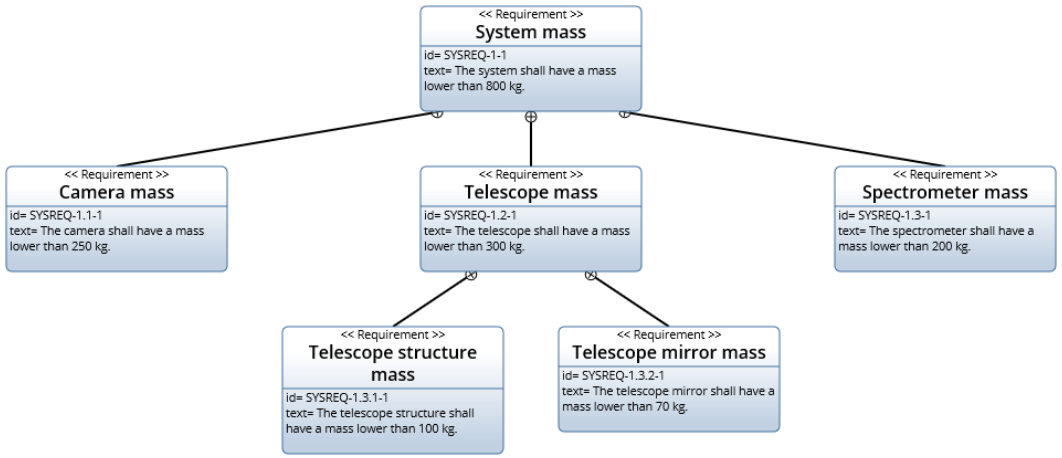
GOOD decomposition

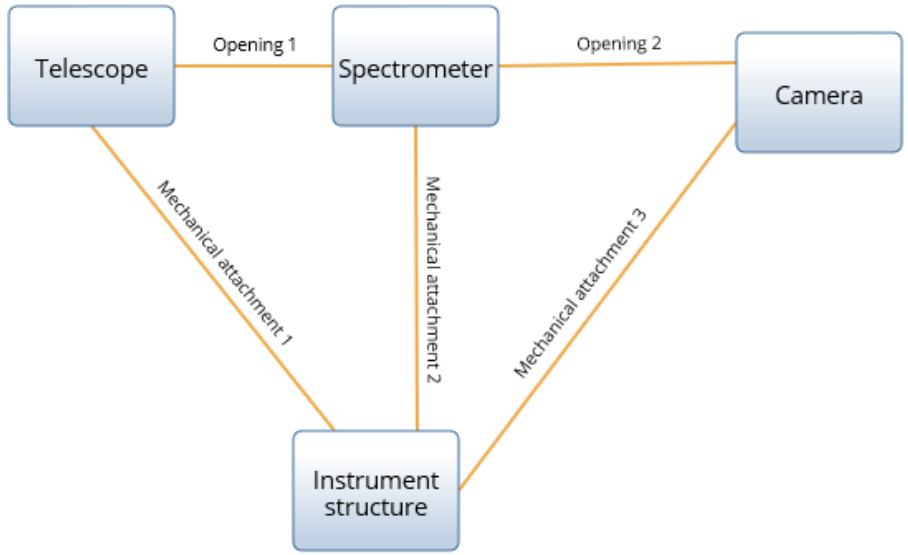
⟷

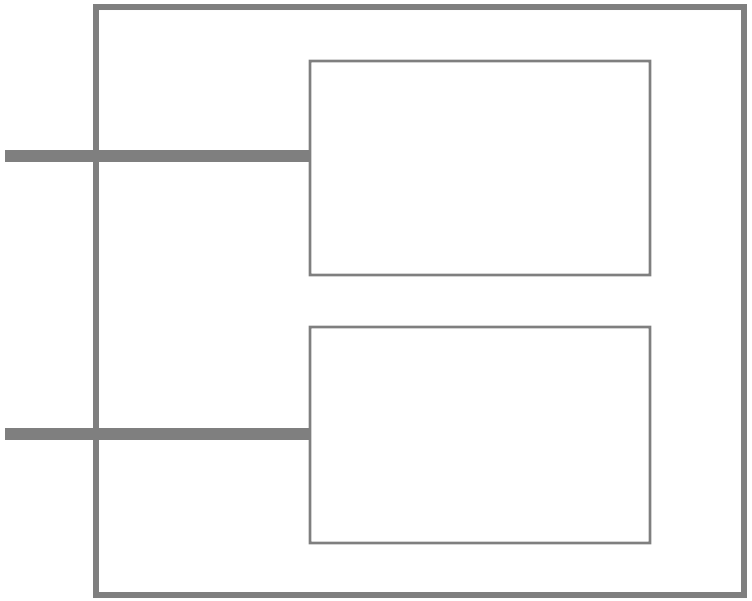
GOOD requirement flow-down

⟷

GOOD interface definition







Comonents do **NOT** have to be **connected** to form another component

A decomposition is valid as long as it is a partition

Remeber the computer, the keyboard, and the key…