SYSTEM-ARCHITECTURE ENGINEERING

2021 COMPETENCE CENTER LEAD

Gilles Ngainsi DEFO – ALTEN Switzerland AG

Competence Center Lead – Engineering – Computer Science

06 | 09 | 2021

Agenda

Agenda

1. ORGANISATIONAL CONTEXT AND DEFINITIONS
2. SYSTEM-ARCHITECTURE ENGINEERING
3. TOOLS & PATTERN FOR SYS- & SW- ENGINEERING
4. QUESTIONS

The Moderator

Dipl.-Ing. Gilles Ngainsi Defo

- **Name:** Gilles Ngainsi Defo
- **Education:** Electrical Engineer – Information Technology Engineer
  Specialist in Telematics and Communication Networks
- **Nationality:** German
- **Languages:** French, English, German, (minimum: Russian, Italian)
- **Current position:** System and Software Engineer
  Competence Centre Lead: Legacy Software
- **Relevant Industries:** Defence, Automotive and Industrial Engineering

(Armasuisse, Schindler, Daimler, BMW, Audi, Volkswagen, Bombardier, TRW Automotive, BOSCH, Automotive Lighting, Eberspächer Controls, Elster, …)


1
ORGANISATIONAL CONTEXT AND DEFINITIONS
ALTEN

ORGANISATIONAL CONTEXT AND DEFINITION

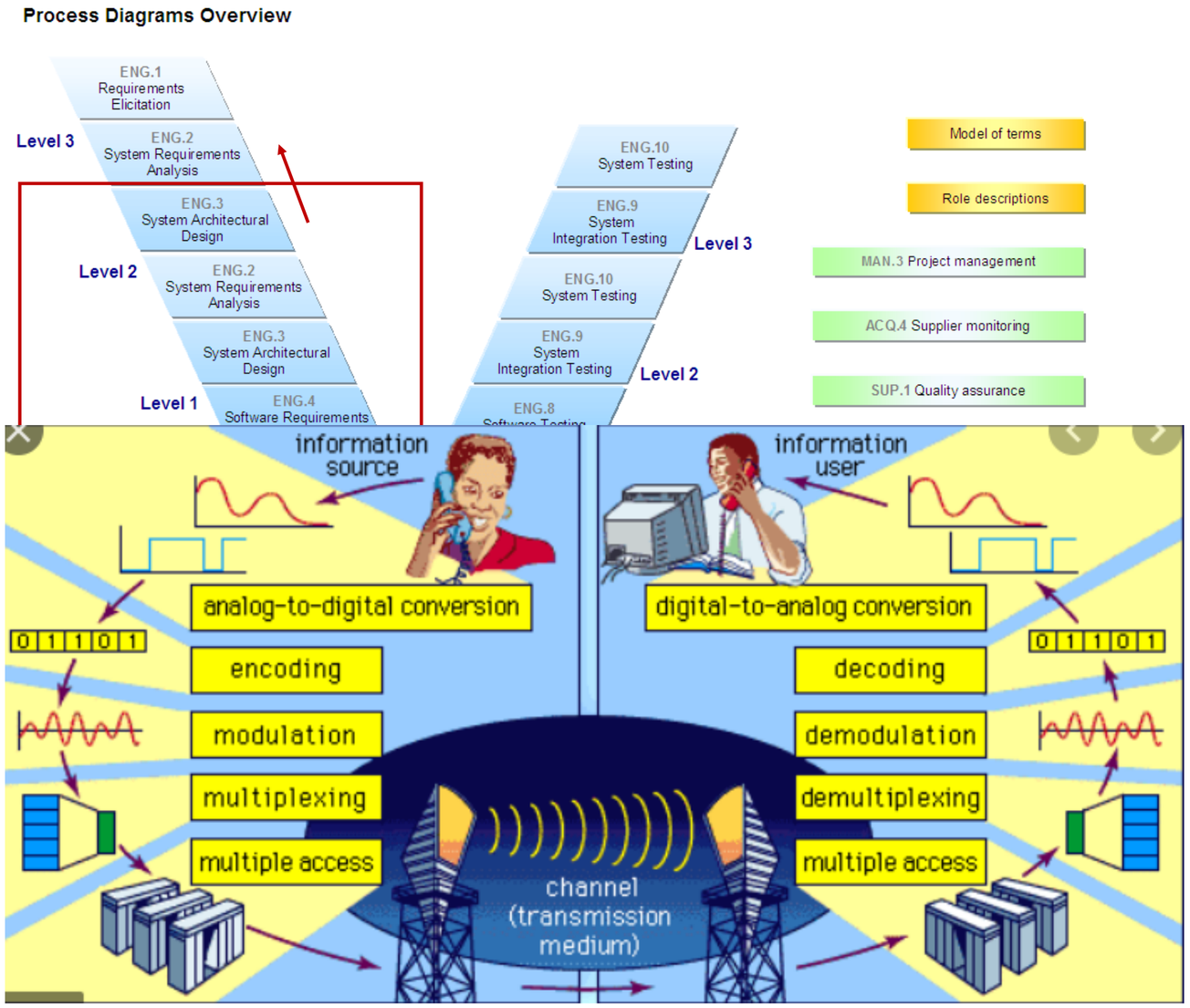
Project Organisation Context and Definitions

- **System:** A telephone or PBX (private branch Exchange)
  => Interconnect people and data (functionality)
  => via voice, video, messaging (quality)
  - with specific expectation: bandwidth (performance)

- **What is Architecture ?**
  "..deals largely with unmeasurables using non-quantitative tools and guidelines based on practical experience knowledge" (Functions, Forms, Interfaces)
  => Inductive processing (heuristic art)

- **What is Engineering ?**
  "..deals with measurables using analytic tools provided by mathematics and hard sciences" (Rechtin, 2002)
  => deductive processing

- **Why do we need System Architecture Engineering?**
  Development of complex engineering systems
  Reuse of elements, components or parts
  Comply with regulations and guidelines
  Ensure overall system quality (and stability)
  Strategic innovations and alternatives



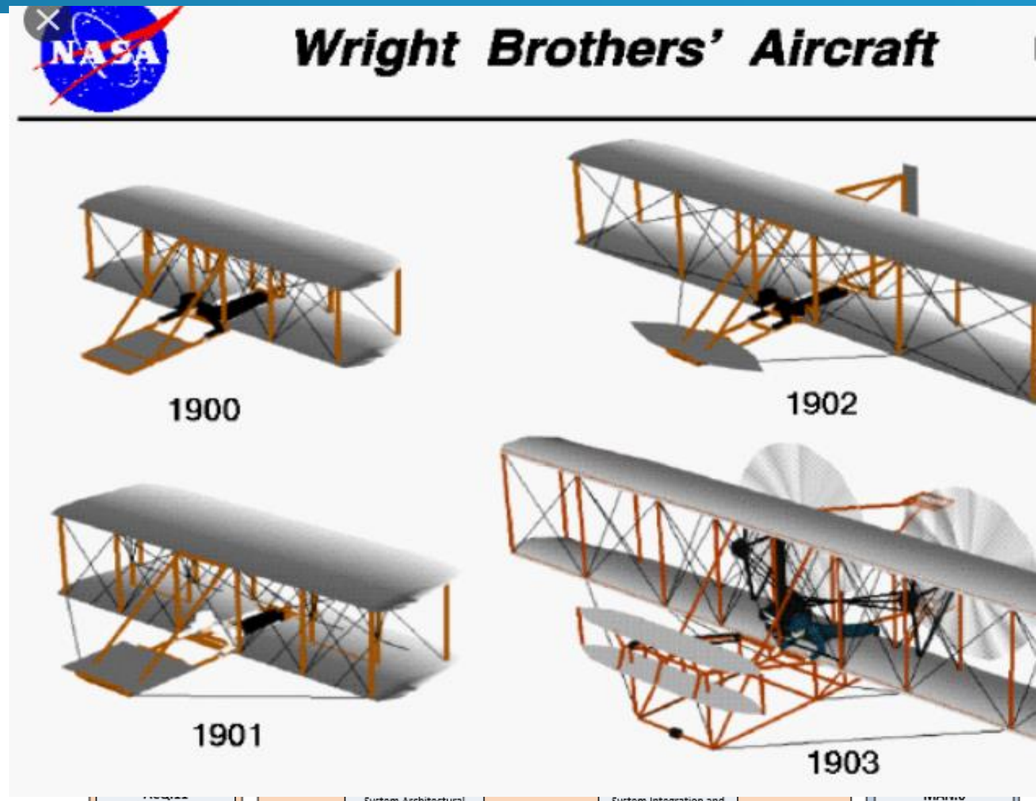
ORGANISATIONAL CONTEXT AND DEFINITION

Example of Properties of System Architecture Engineering

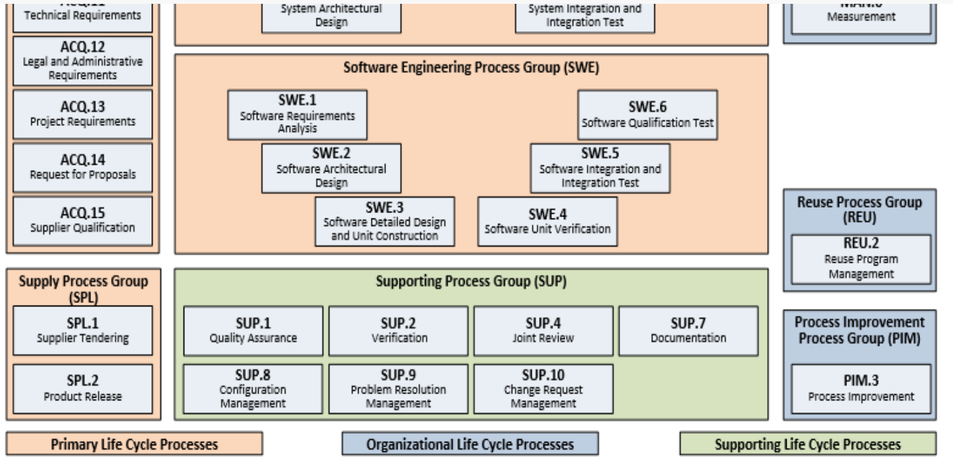
- **Why do we need System Architecture Engineering?**
  - Development of complex engineering systems
  - Reuse of elements, components or parts
  - Comply with regulations and guidelines
  - Ensure overall system quality (and stability)
  - Strategic innovations and alternatives

- **Wright brothers' original aeroplane or aircraft**
  - Less complexity (materials: wood and paper, motor)
  - Less quality (only one passenger; lying position)
  - Less performance (scalability in reach, size, capacity)
  - Limited safety (pressure, weather)

=> Architecture has limitations
=> Scaling up the system led to dramatical complexity
=> System Architecture Engineering necessary !

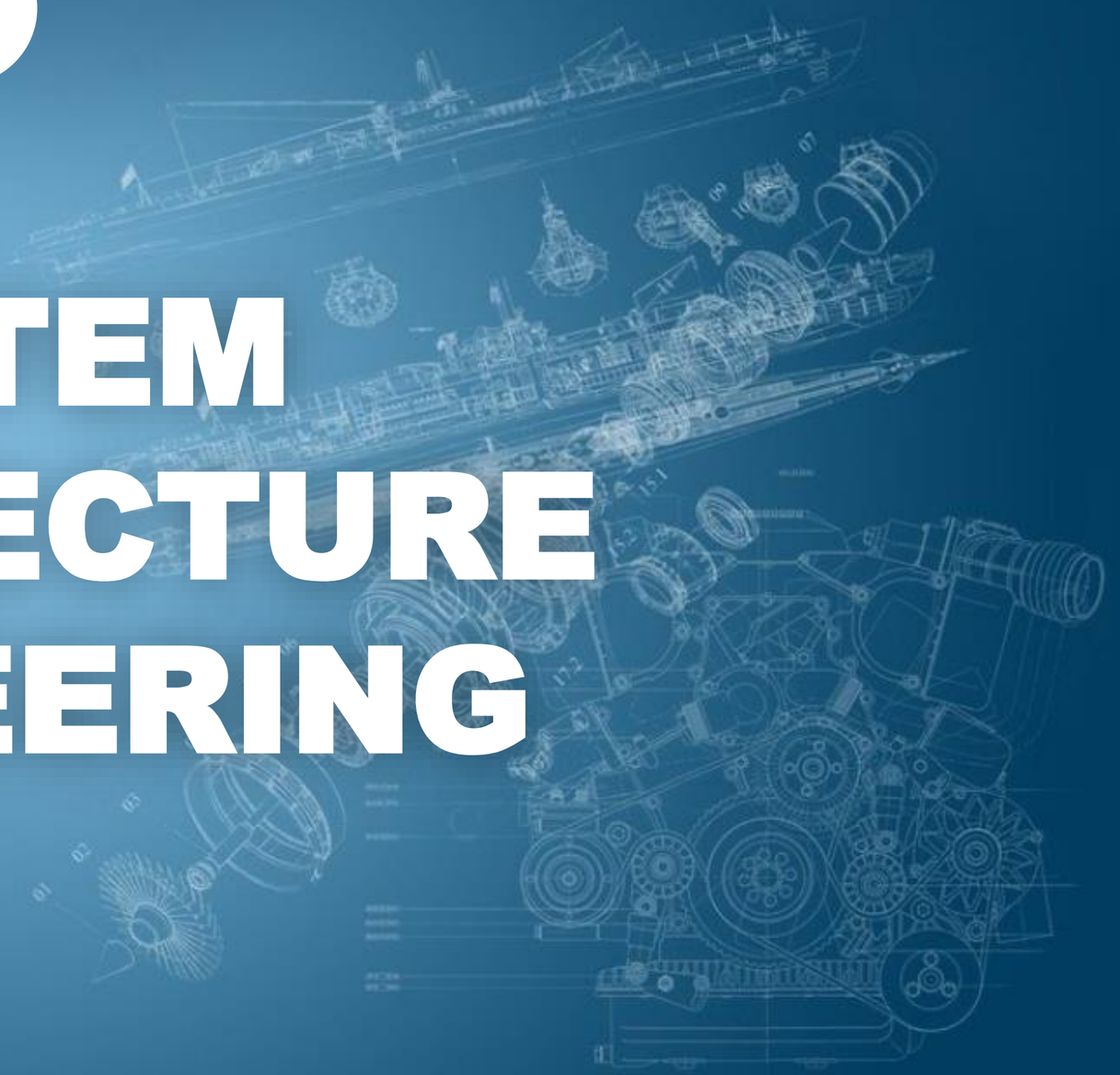


Source: NASA / Glenn Research Center



2

SYSTEM ARCHITECTURE ENGINEERING



SYSTEM-ARCHITECTURE ENGINEERING

Managing complexity

- **Decomposition**
  - Detaching the system in parts
  - Refining system parts into elements

  => Parts, Elements: easier to understand, manage

- **Abstraction**
  - Assigning Elements characteristics:
    - Properties (Methods or Functions)
    - Attributes (parameters)
    - Interfaces (Interconnections for exchange)

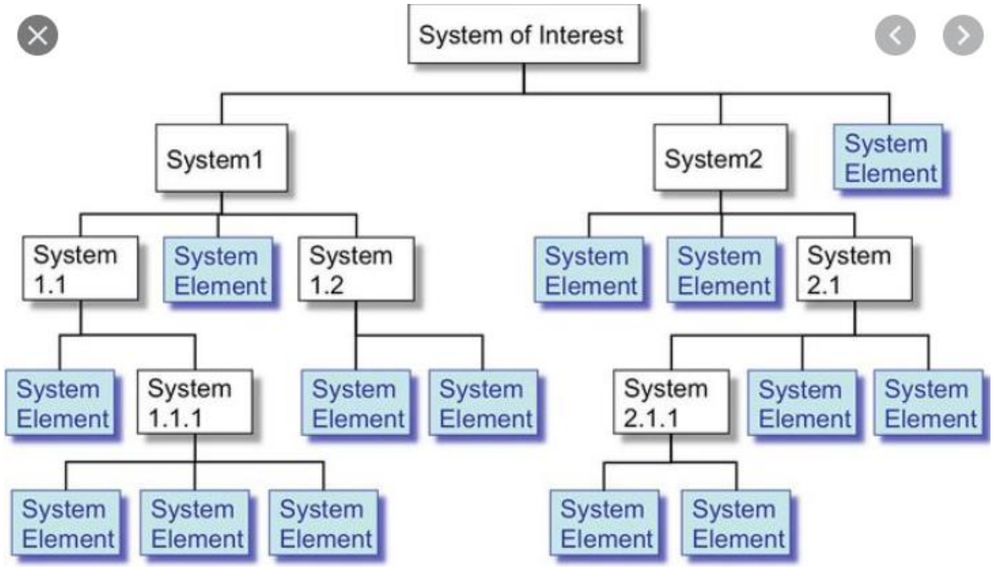
  => Ground for modelling an element

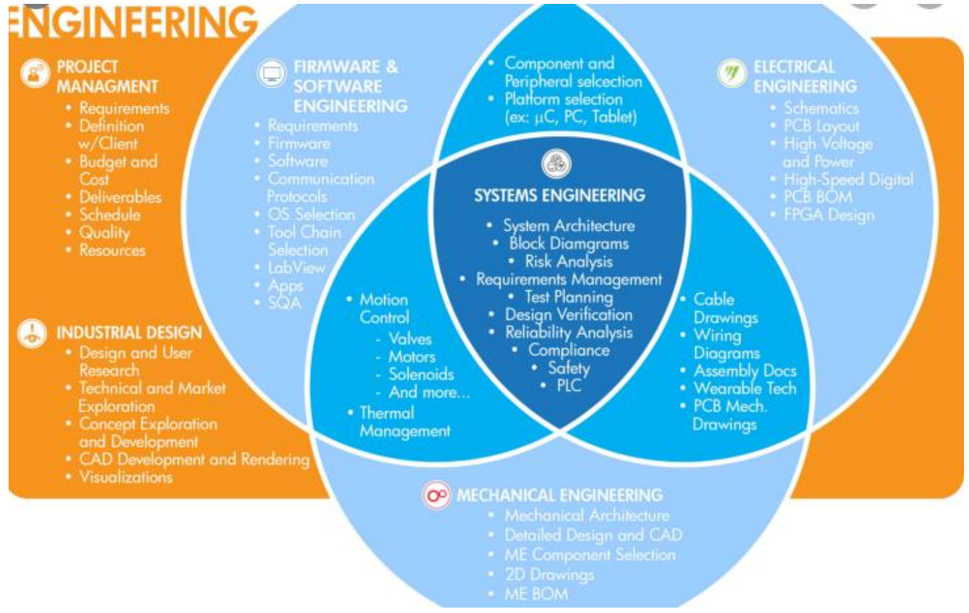
- **Hierarchy**
  - Organize elements according to a scheme/order
    - Sustains and makes use of decomposition: structure
    - Complementary contribution to System Integration
    - Focus on relevance (design orientations)

  ⇒ Handles priorities and builds up the levels (layers)

  => Hardware – Soft-&Firmware – Mechanical domains

  => Functional – non-Functional





- **Technology**
  Target Product functionality and performance aspects

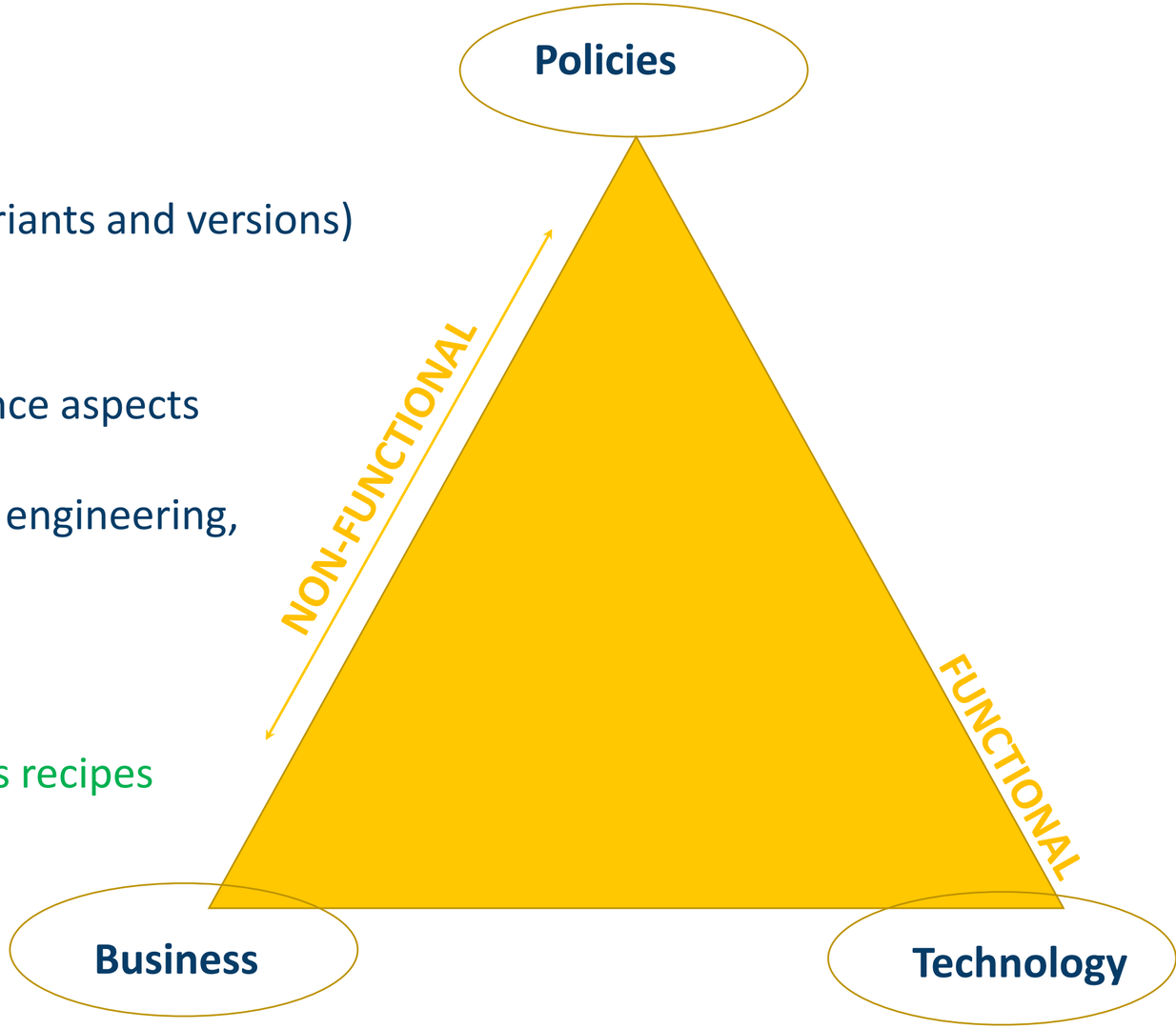
- **Business**
  Costs and marketing driven aspects (growth, sales, scalability, variants and versions)

- **Policies**
  - Regulations and internationals standards and norms compliance aspects
  - Restrictions or constraints

  (time, branch, geographic location, domains: e.g. Military or civil engineering, SAE or CE)

- **Since architecture is a heuristic art:**
  => There is no exact solution but approximate solution
  => Often design patterns are involved or even raised as standards recipes

...

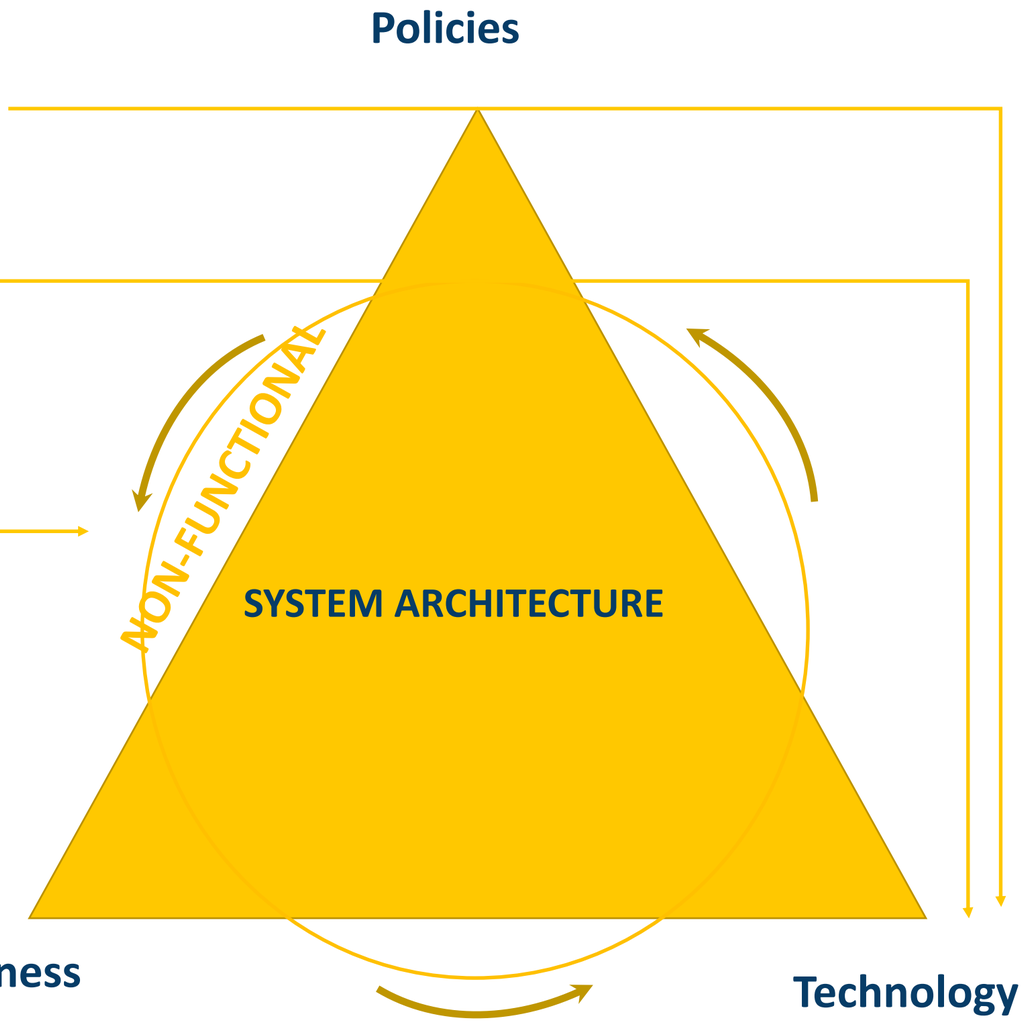


SYSTEM-ARCHITECTURE ENGINEERING

System and Software architecture engineering

SYSTEM (or SOFTWARE) ARCHITECTURE IS … :

- How the defining components of a system are assembled and organized
  - ✓ Architectural pattern of the system
- How those components are interacting with each other
  - ✓ Interfaces, junctions, mounting points
  - ✓ (messaging, API)
- the constraints the whole system is submitted to
  - ✓ Non-functional (Quality attributes: “-ities”)
- Architecture is the input of the development phase



… MORE THAN JUST THE OVERALL PATTERN OR STRCTURE !

SYSTEM-ARCHITECTURE ENGINEERING

Breaking Down Projects Activities: System Architecture Design

- **Define System Level 3 Architecture**
  Decomposition, Hierarchy, Abstraction
  => Physical model (system parts and elements)

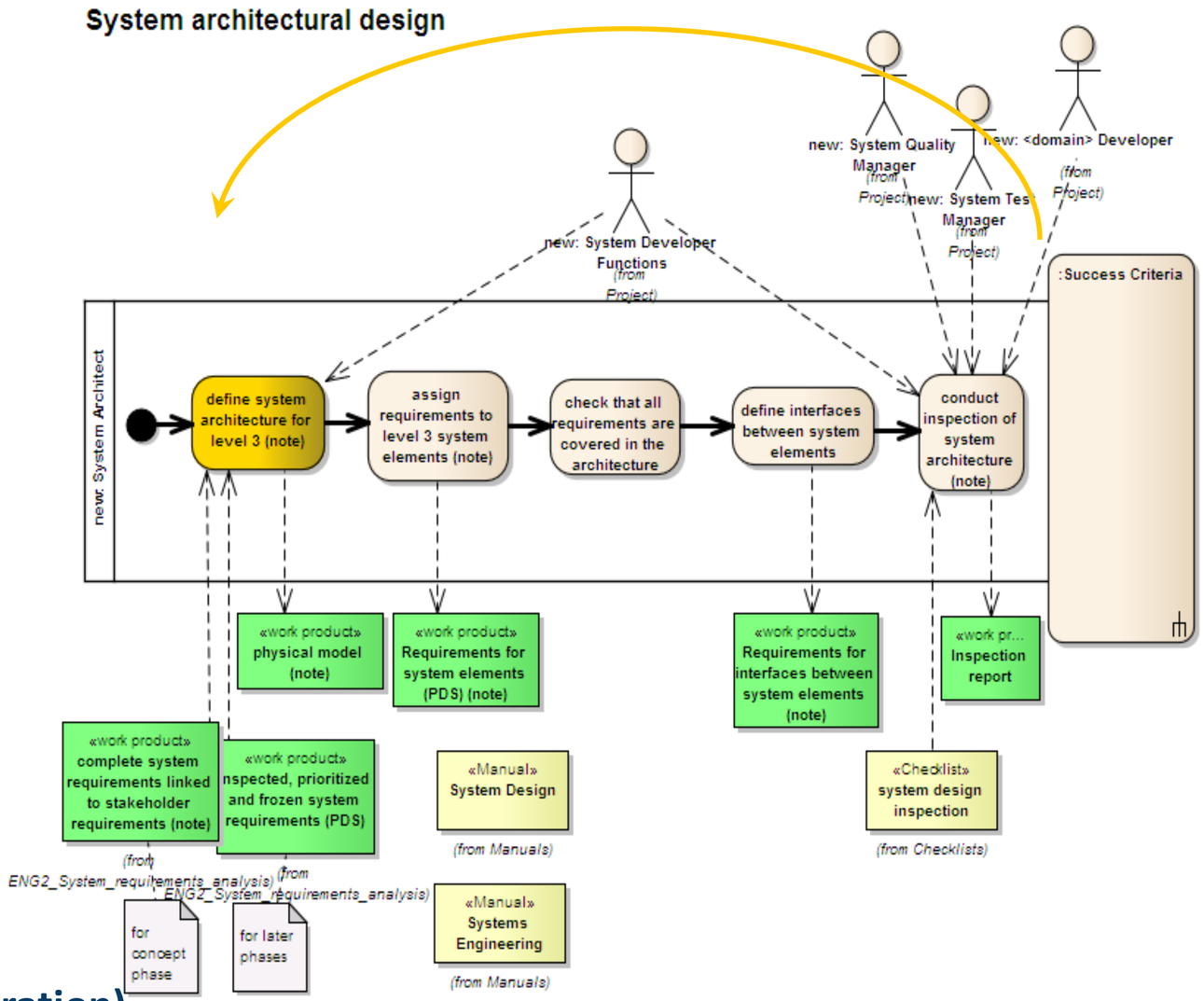
- **Assign requirements to level 3 elements**
  Linking for traceability and validation
  => Requirements for system elements

- **check requirements coverage by architecture**
  Verify requirements coverage through elements
  => Create Design System and elements Diagrams

- **define interfaces between elements**
  Model the system functionality by interactions
  => Simulate the diagrams

- **Review the outcome in Team**
  Inspect and review the system architecture
  Conduct FMEA and Design Validation
  => Update elements and interfaces and design

- **Repeat the steps for System Level 2 & 1 Architecture (iteration)**



SYSTEM-ARCHITECTURE ENGINEERING

Modeling Architecture and Design elements

- **How do we define and design elements and interactions?**
  - Use of a perspective approach (IREB ©)
  - Model the design with uniform standardized tools and language (UML)
  - Inheritance from Requirements Engineering

  => Ensures coverage and traceability of architecture to requirements

- **structure perspective**
  - input and output data
  - Static aspects of use and interdependency relationship between elements

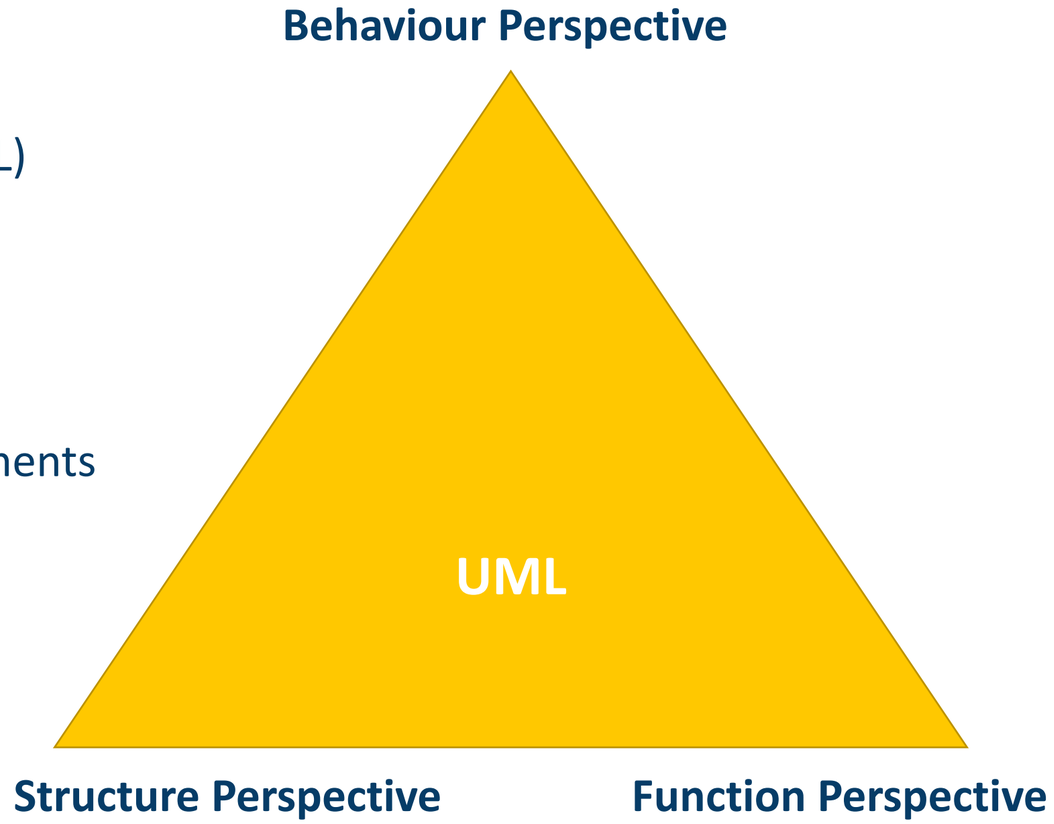
  => Mostly the starting approach for Level 3 (UC-, ER-, Class-Diagrams)

- **Function perspective**
  - Model the system functionality by information flows via interfaces
  - Information transformation between systems and context is focused

  => Simulation (Data flow-, Activity Diagrams)

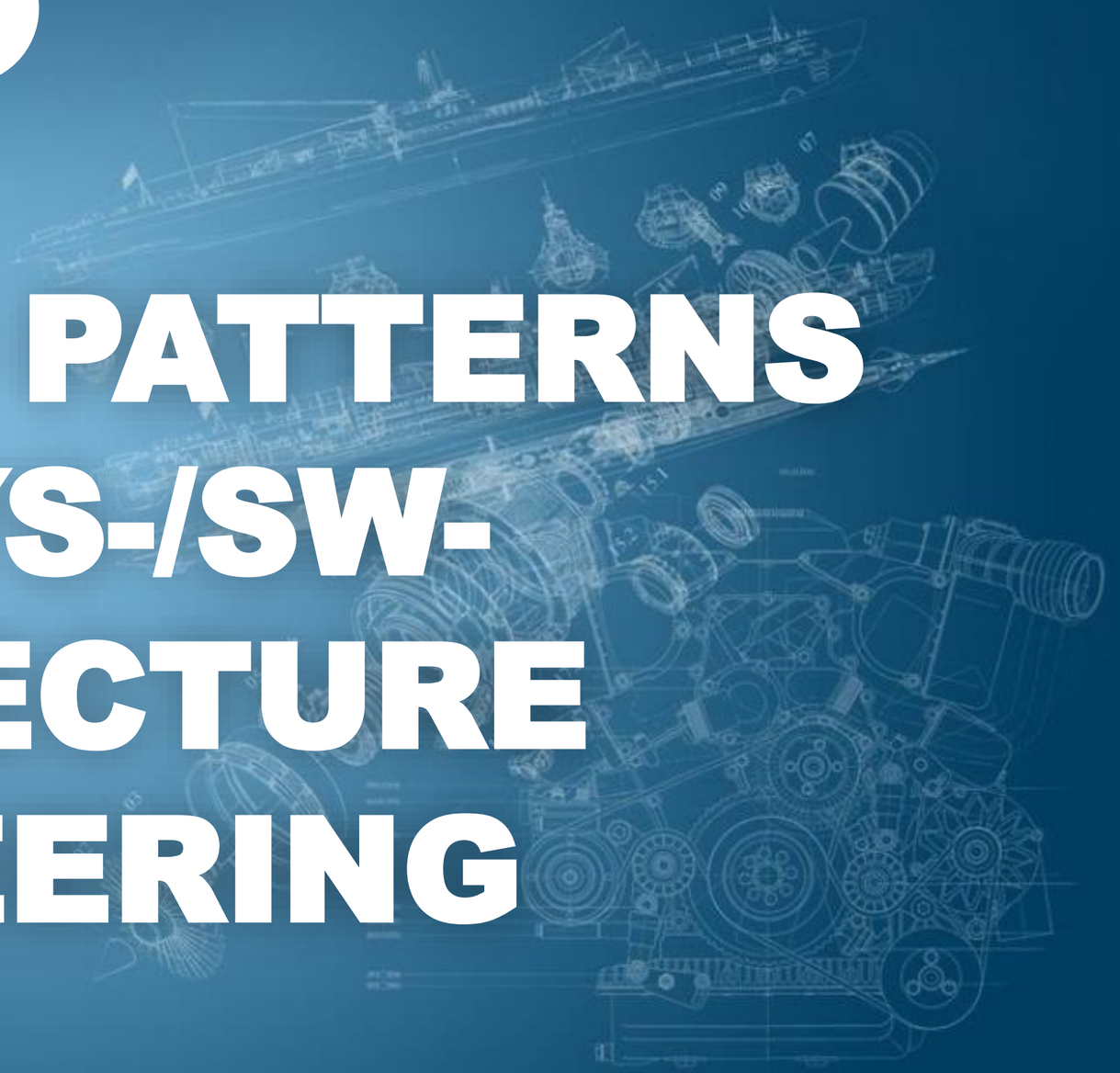
- **Behaviour perspective**
  - Best suited for event driven systems (Information systems)
  - Systems states are placed in focus and triggered by events and conditions

  => Simulation (State Charts-, Sequence Diagrams)



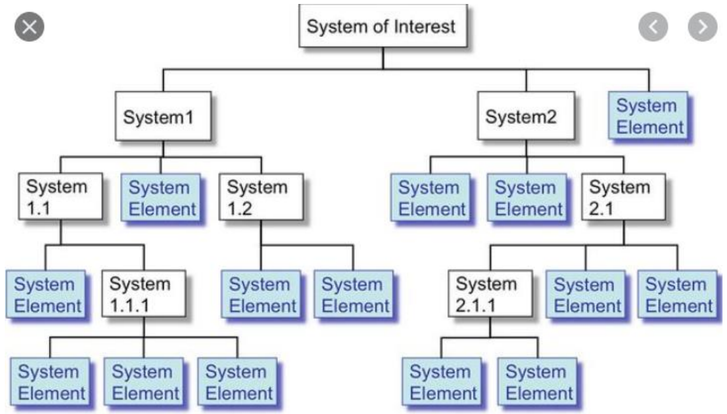
3

TOOLS AND PATTERNS FOR SYS-/SW-ARCHITECTURE ENGINEERING



Patterns: Definition

ARCHITECTURAL PATTERN MEANS:



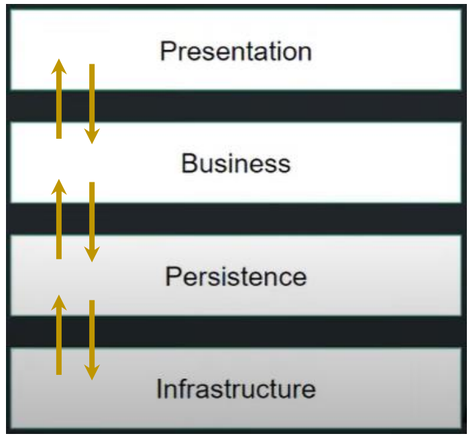
- **The granularity of the components building the architecture**
  - ✓ Granularity: how small or big the component is designed
- **How detailed are component defined?**
  - ✓ Would be there few coarse (refinable) components? (Layered architectural patterns)
  - ✓ Or very small independent components from start? (Microservices architectural patterns)
- **The Architectural pattern will help us make good decisions for development phase**
  - ✓ Design patterns
- **Difference between the Architectural Pattern and the Design Pattern?**
  - ✓ Design Pattern: how components are individually built
  - ✓ Design Pattern: low level scope
  - ✓ Architectural Pattern: how the components are organized and assembled
  - ✓ Architectural Pattern: high-level scope
- **The Architectural pattern affects the choice of Design Pattern**

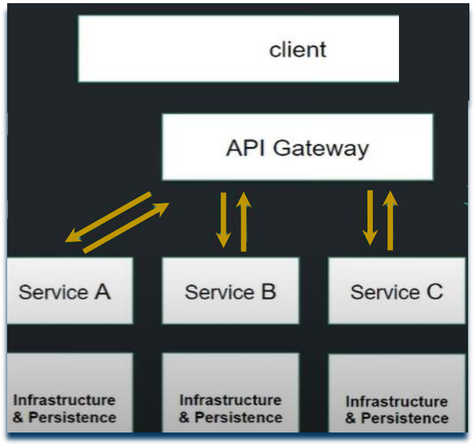
Architectural Patterns Types

ARCHITECTURAL PATTERN TYPES:

Level 1	Monolithic	Service-based	Distributed
Level 2	Layered	Microservices	Event-Driven
	...	...	...

Examples





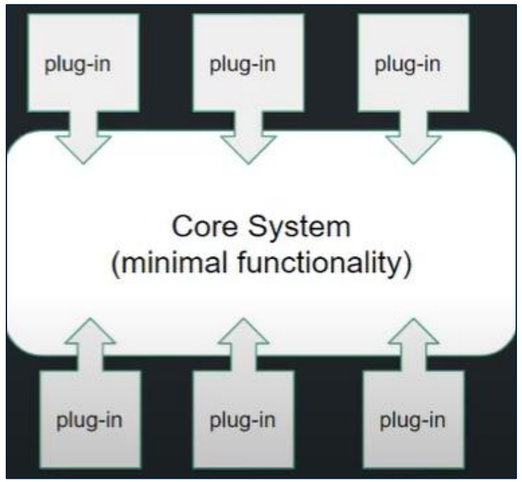
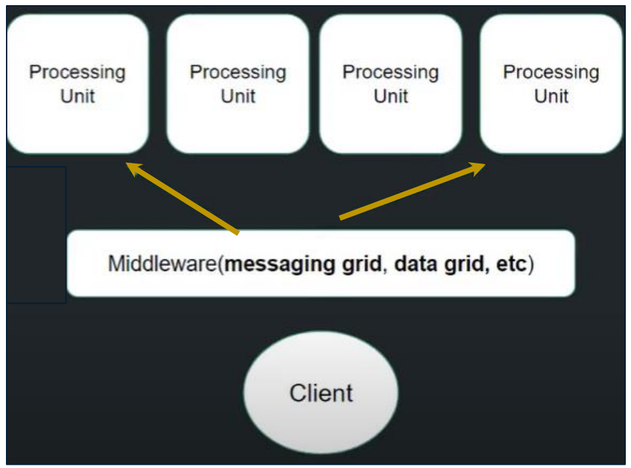
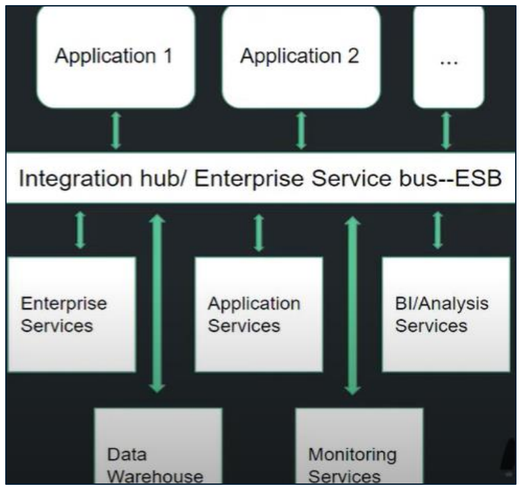
- ✓ Most used is SW Systems
- ✓ Separation of concerns
- ✓ Layers as components

- ✓ Is NOT service oriented
- ✓ System doesn't break by adding/removing
- ✓ Independently evolvable deployable units

- ✓ Event processing units coordinated by Dispatcher
- ✓ Asynchronous, dynamic highly scalable
- ✓ Structure depends on topology

Architectural Patterns Types

ARCHITECTURAL PATTERN TYPES:

Level 1	Monolithic	Hybrid	Hybrid
Level 2	Microkernel	Space-based	Services-Oriented
Examples			

- ✓ Plug-ins add functionality
- ✓ Plugins add customization
- ✓ Web Browsers, Text Editors, OS

- ✓ Distributed caching – In memory Data Grid
- ✓ Elasticity and high-scalability,
- ✓ Computer Systems, Cloud or Social Media

- ✓ Built upon each other
- ✓ Large enterprise systems
- ✓ Coarse-grained and large scope of use

TOOLS AND PATTERNS FOR SYSTEM & SOFTWARE ARCHITECTURE ENGINEERING

Tools in System and Software Architecture modeling

MODELING TOOLS:

- **Enterprise Architect (Sparx Systems, 1998): System & Software**
  - ✓ Well suited for all 3 languages
  - ✓ Allows simulation of functionality through diagrams
  - ✓ Supports scripting and code generation
  - ✓ Best for Software
- **Rhapsody (I-Logix / IBM, 1998): System & Software**
  - ✓ Well suited for all 3 languages
  - ✓ Allows simulation of functionality through diagrams
  - ✓ Supports scripting and code generation
  - ✓ Best for Software
- **CATIA (Dassault Systèmes, 1982): System & Hardware/Mechanics**
  - ✓ Focused for 3D-Design
  - ✓ Suited for Hardware material and Mechanical parts
  - ✓ Available as Cloud version since 2014
- **OrCAD Spice (ORCAD Sys. Corp., 1985): Hardware**
  - ✓ 3D/2D-Design
  - ✓ Suited for Hardware material and electronics circuitry
  - ✓ Captures, Simulates and Layouts printed Circuit Boards

MODELING LANGUAGES:

- **UML (Unified modeling language)**
- **SysML (System Modeling language)**
- **BPML (Business Process Modeling language)**

SYSTEM ACHITECTURE REFERENCES:

- “The art of System Architecture” (Rechtin 2002)
- “Augustine’s Laws” (Norman Austin 1987)
- “System Engineering mit SysML/UML” (Tim Weilkiens 2008)

...


Thank you for your attention !
Questions?