Physical Architecture Model Development

From SEBoK
Physical Architecture Model Development

Physical Architecture Model Development may be used as a task of the activity "Develop candidate architectures models and views", or a sub-process of the System Architecture Definition process (see **System Architecture** article).Its purpose is to elaborate models and views of a physical, concrete solution that accommodates the logical architecture model and satisfies and trades-off system requirements. Once a logical architecture model is defined (see Logical Architecture Model Development), concrete physical elements have to be identified that can support functional, behavioral, and temporal features as well as the expected properties of the system deduced from non-functional system requirements (e.g. constraint of replacement of obsolescence, and/or continued product support).

A physical architecture model is an arrangement of physical elements, (system elements and physical interfaces) that provides the solution for a product, service, or enterprise. It is intended to satisfy logical architecture elements and system requirements ISO/IEC/IEEE 26702 (ISO 2007). It is implementable through technological system elements. System requirements are allocated to both the logical and physical architectures. The resulting system architecture is assessed with system analysis and when completed becomes the basis for system realization.

In some cases, particularly when multiple systems are to be defined to a common physical architecture model, one of the drivers for the physical architecture model may be interface standards; these physical interfaces may well be one of the most important concerns for these systems. It is quite possible that such interface standards are mandated at a high level in the system requirements. On the other hand, it is equally possible for standards to be derived during physical architecture model development and these can be critical enablers for desirable engineering outcomes, such as: families of systems, technology insertion, interoperability and “open systems”. For example, today’s video, hi-fi, and computer systems have all benefited from adoption of interface standards. Other examples exist in most fields of engineering from nuts and bolts, plumbing, electrical installations, rail gauges, TCP/IP, IT systems and software to modular defense and space systems.

Note: The term *Physical Architecture* is a contraction of the expression *Physical View of the System Architecture*.

Contents

Concepts and Principles

System Element, Physical Interface, and Physical Architecture Model

A system element is a discrete part of a system that can be implemented to fulfill design properties. A system element can be hardware, software, data, humans, processes (e.g., processes that provide a service to users), procedures (e.g., operator instructions), facilities, materials, and naturally occurring entities (e.g., water, organisms, and minerals), or any combination of these ISO/IEC/IEEE 15288 (ISO 2015). A physical interface binds two system elements together; this is similar to a link or a connector. Table 1 provides some examples of system elements and physical interfaces.

Table 1. Types of System Elements and Physical Interfaces. (SEBoK Original)

Element	Product System	Service System	Enterprise System
System Element	• Hardware Parts (mechanics, electronics, electrical, plastic, chemical, etc.) • Operator Roles • Software Pieces	• Processes, Data Bases, Procedures, etc. • Operator Roles • Software Applications	• Corporate, Direction, Division, Department, Project, Technical Team, Leader, etc. • IT Components
Physical Interface	* Hardware Parts, Protocols, Procedures, etc.	* Protocols, Documents, etc.	* Protocols, Procedures, Documents, etc.

A complex system composed of thousands of physical and/or intangible parts may be structured in several layers of systems and system elements. The number of elements in a level of the structure of one system is limited to only a few, in order to facilitate managing the system definition; a common guideline is *five plus or minus two* elements (see illustration in Figure 1).

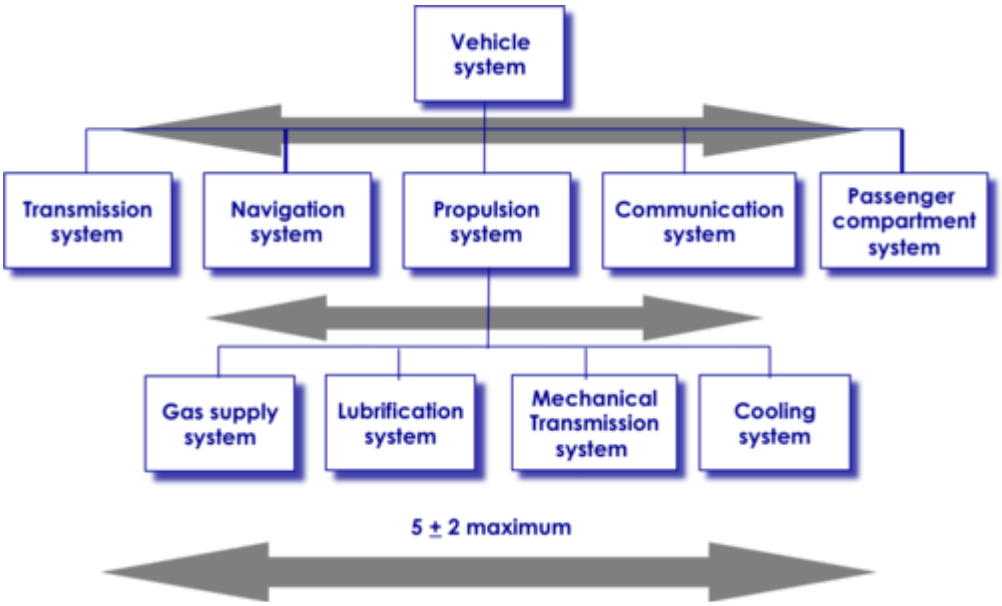

Figure 1. Layers of Systems and System Elements (Faisandier 2012).

A physical architecture model is built from systems, system elements, and all necessary physical interfaces between these elements, as well as from external elements (neighboring or enabling systems and/or system elements in the considered layer and concerned elements in the context of the global system-of-interest) - see illustration in Figure 2.

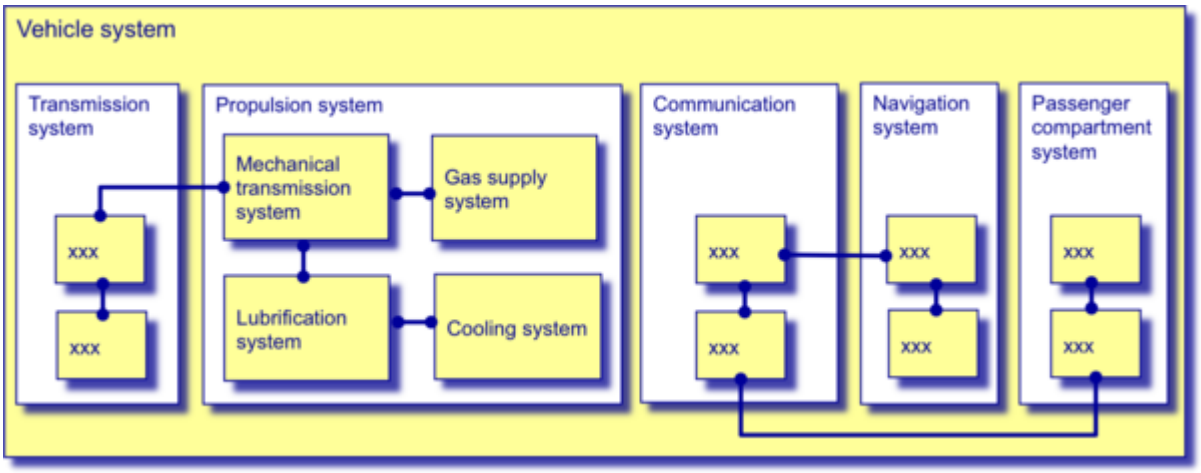

Figure 2. Physical Architecture Model Representation (Faisandier 2012).

Design Property

A design property is a property that is obtained during system architecture and created through the assignment of non-functional requirements, estimates, analyses, calculations, simulations of a specific aspect, or through the definition of an existing element associated with a system element, a physical interface, and/or a physical architecture. If the defined element complies with a requirement, the design property will relate to (or may equal) the requirement. Otherwise, one has to identify any discrepancy that could modify the requirement or design property, and detect any deviations.

Stakeholders have concerns that correspond to the expected behavior of a system within operational, environmental, and/or physical constraints as well as to more general life cycle constraints. Stakeholder requirements and system requirements express these concerns as expected capabilities from the system (e.g., usability, interoperability, security, expandability, environment

suitability, etc.). Architects and/or designers identify these capabilities from requirements and deduce corresponding quantitative or qualitative design properties to properly equip their physical architecture model (e.g., reliability, availability, maintainability, modularity, robustness, operability, climatic environment resistance, dimensions limits, etc.). For further discussion on how some of these properties may be included in architecture and design, please see the article Systems Engineering and Specialty Engineering in the Related Disciplines knowledge area (KA).

Allocation of Logical Elements to Physical Elements and Partitioning

Developing a candidate physical architecture model for a system consists of first identifying the system elements that can perform functions of the logical architecture model as well as identifying the interfaces capable of carrying out the input-output flows and control flows. When identifying potential elements, a systems engineer needs to allocate design properties within the logical architecture; these properties are deduced from the system requirements. Partitioning and allocation are activities to decompose, gather, or separate functions in order to facilitate the identification of feasible system elements that support these functions. Either they exist and can be reused or re-purposed, or they can be developed and technically implemented.

Partitioning and allocation use criteria to find potential affinities between functions. Systems engineers use system requirements and/or design properties as criteria to assess and select candidate system elements and partitions of functions, such as similar transformations within the same technology, similar levels of efficiency, exchange of the same type of input-output flows (information, energy, and materials), centralized or distributed controls, execution with close frequency level, dependability conditions, environment resistance level, and other enterprise constraints.

A concurrent engineering approach is necessary when several different sets of technologies, knowledge, and skills are necessary to establish a candidate physical architecture model. This is particularly true during the partition and allocation of functions to various system elements, in which the systems engineer must account for compatibility issues and emergent properties. (see SEBoK Part 2: Synthesizing Possible Solutions for a discussion of possible approaches)

Developing Candidate Physical Architecture Models

The goal of physical architecture model development activities is to provide the best possible physical architecture model made of suitable systems, technological system elements, and physical interfaces (i.e., the architecture that answers, at best, all system requirements, depending on agreed limits or margins of each requirement). The best way to do this is to produce several candidate physical architecture models, assess and compare them, and then select the most suitable one.

A candidate physical architecture model is elaborated according to affinity criteria in order to build a set of system elements (i.e., separate, gather, connect, and disconnect the network of system elements and their physical interfaces). These criteria are the same as those used for partitioning and allocating functions to system elements. The physical architecture model development may be focused in different ways, for example, it may address:

- Reduction in the number of physical interfaces
- System elements that can be tested separately
- Compatible technology
- Measures of the proximity of elements in space
- Ease of handling (weight, volume, and transportation facilities)
- Optimization of resources shared between elements
- Modularity (i.e. elements have low interdependence)
- Resilience (i.e. elements which are highly reliable, maintainable or replaceable)

Evaluating and Selecting the Preferred Candidate

Viable physical architecture models enable all required functions or capabilities specified in the logical architecture model to be realized. Architecture and design activity includes evaluation to obtain a balance among design properties, costs, risks, etc. Generally, the physical architecture model of a system is determined more strongly by non-functional requirements (e.g., performance, safety, security, environmental conditions, constraints, etc.) than by functions. There may be many (physical) ways to establish functions but fewer ways of satisfying non-functional requirements. The preferred physical architecture model represents the selection of system elements, their physical relationships, and interfaces. Typically this physical architecture will still leave further systems engineering to be undertaken to achieve a fully optimized system after any remaining trade-offs are made and algorithms and parameters of the system are finalized.

Certain analyses (efficiency, dependability, cost, risks, etc.) are required to get sufficient data that characterize the global behavior and structure of the candidate architectures in regard to system requirements; this is often broadly referred to as system analysis. Other analyses and assessments require knowledge and skills from the different involved technologies and specialities (mechanics, electronics, software, thermodynamics, electro-magnetic compatibility, safety, security etc.). They are performed through corresponding specialist analysis of the system.

Legacy Systems and Systems of Systems

Few systems come into existence or operate without interacting with others in a system context. These interactions may be with other operational systems, or maintenance and support systems, which in turn may be legacy (already in use) or future legacy (under development and likely to operate with the system of interest in the future).

The best chosen approach will be dependent on the strength of interactions between the system-of-interest (SoI) and its wider context. While these interactions are small, they may be accounted for by defining a set of static external interface requirements (for example technical standards) with which the system must comply, by including these as constraints in the system requirements and ensuring compliance through design assurance.

Where the interactions are more intense, for example where continuous information is to be exchanged with other systems, these will have to be recognized as part of a system of systems context and will instead be considered as part of an enterprise systems engineering approach.

Another important consideration may be the sharing of technology or system elements between the SoI and other systems, often as part of a family of systems (many examples occur in automotive and aerospace industries) or the re-use of system elements from existing legacy. Here a degree of top-down or middle-out design work will be necessary to ensure the system of interest embodies the required system elements, while conforming as far as possible to the stakeholder and system requirements, with any compromises being understood and managed.

If a System-of-Interest is intended to be used in one or more service systems or system of systems configurations this will affect its physical architecture model. One of the features of these SoS is the late binding of component systems in use. Such component systems must be architected with open or configurable interfaces, must have clearly defined functions packaged in such a way as to be relevant to the SoS using them, and must include some method by which they can be identified and included in the SoS when needed.

Both service systems and SoS will be defined by a high level physical architecture model, which will be utilized to define the relevant SoS relationships, interfaces, and constraints that should be included in Concept Definition. The results will be embedded in the stakeholder and system requirements and handled through interface agreements and across-project communication during development, realization, and use.

See SEBoK Part 4 Applications of Systems Engineering for more information on special considerations for architecting SoS.

Process Approach

Purpose

The purpose of the Physical Architecture Model Development is to define, select, and synthesize a system physical architecture model which can support the logical architecture model. A physical architecture model will have specific properties to address stakeholder concerns or environmental issues and to satisfy system requirements.

Because of the evolution of the context of use or technological possibilities, the physical architecture which is composed of system elements is supposed to evolve along the life cycle of the system in order for it to continue to perform its mission within the limits of its required effectiveness. Depending on whether or not evolution impacts logical architecture model elements, allocations to system elements may change. A physical architecture model is equipped with specific design properties (glossary) to continuously challenge the evolution.

Generic inputs include the selected logical architecture model, system requirements, generic patterns and properties that architects identify and utilize to answer requirements, outcomes from system analysis, and feedback from system verification and system validation.

Generic outputs are the selected physical architecture model, allocation matrix of functional elements to physical elements, traceability matrix with system requirements, stakeholder requirements of each system and system element composing the physical architecture model, and rejected solutions.

Activities of the Process

Major activities and tasks to be performed during this process include the following:

- Partition and allocate functional elements to system elements:
  - Search for system elements or technologies able to perform functions and physical interfaces to carry input-output and control flows. Ensure system elements exist or can be engineered. Assess each potential system element using criteria deduced from design properties (themselves deduced from non-functional system requirements).
  - Partition functional elements (functions, scenarios, input-outputs, triggers, etc.) using the given criteria and allocate partitioned sets to system elements (using the same criteria).
  - When it is impossible to identify a system element that corresponds to a partitioned functional set, decompose the function until the identification of implementable system elements is possible.
  - Check the compatibility of technologies and the compatibility of interfaces between selected system elements.
- Constitute candidate physical architecture models.
  - Because partitioned sets of functions can be numerous, there are generally too many system elements. For defining controllable architectures, system elements have to be grouped into higher-level system elements known as system element groups, often called sub-systems in industry.
  - Constitute several different system element groups corresponding to different combinations of elementary system elements. One set of system element groups plus one or several non-decomposable system elements form a candidate physical architecture model of the considered system.
  - Represent (using patterns) the physical architecture model of each system element group

connecting its system elements with physical Interfaces that carry input-output flows and triggers. Add physical interfaces as needed; in particular, add interfaces with external elements to the system element group.
    - Represent the synthesized physical architecture of the considered system built from system element groups, non-decomposable system, and physical interfaces inherited from the physical architecture model of system element groups.
    - Enhance the physical architecture model with design properties such as modularity, evolution capability, adaptability to different environments, robustness, scalability, resistance to environmental conditions, etc.
    - If possible, use executable architecture prototypes (e.g., hardware-software (HW-SW)-in-the-loop prototypes) for identifying potential deficiencies and correct the architecture as needed.
- Assess physical architecture model candidates and select the most suitable one:
    - Use the system analysis process to perform assessments (see the System Analysis topic).
    - Use the Decision Management process to support the trades and selection of the preferred alternative (see the Decision Management topic).
- Synthesize the selected physical architecture model:
    - Formalize physical elements and properties. Verify that system requirements are satisfied and that the solution is realistic.
    - Identify the derived physical and functional elements created for the necessity of architecture and design and the corresponding system requirements.
    - Establish traceability between system requirements and physical elements as well as allocate matrices between functional and physical elements.

Artifacts, Methods and Modeling Techniques

Physical architecture descriptions use modeling techniques to create and represent physical architectures. Some common physical models include structural blocks, mass, layout and other models. Modeling techniques may be:

- Physical block diagrams (PBD)
- SysML block definition diagrams (BDD)
- Internal block diagrams (IBD) (OMG 2010)
- Executable architecture prototyping
- Etc.

Depending on the type of domain for which it is to be used (defense, enterprise, etc.), architecture frameworks may provide descriptions that can help to trade-off candidate architectures. Please see section 'Enterprise Architecture Frameworks & Methodologies' in Enterprise Systems Engineering Key Concepts.

Practical Considerations

Key pitfalls and good practices related to physical architecture development are described in the next two sections.

Pitfalls

Some of the key pitfalls encountered in performing physical architecture model development are provided in Table 3.

Table 3. Pitfalls with Physical Architecture Development. (SEBoK Original)

Pitfall	Description

Too Many Levels in a Single System Block	The current system block includes too many levels of decomposition. The right practice is that the physical architecture model of a system block is composed of one single level of systems and/or system elements.
No Logical Architecture Model	The developers perform a direct passage from system requirements to physical architecture model without establishing a logical architecture model; this is a common wrong practice that mainly takes place when dealing with repeating systems and products because the functions are already known. The issue is that a function is always associated with input-output flows defined in a specific domain set. If the domain set changes, the performance of the function can become invalid.
Direct Allocation on Technologies	At a high level of abstraction of multidisciplinary systems, directly allocating the functions onto technologies of the lowest level of abstraction, such as hardware or software, does not reflect a system comprehension. The right practice is to consider criteria to decompose the architecture into the appropriate number of levels, alternating logical and physical before reaching the technology level (the last level of the system).

Proven Practices

Some proven practices gathered from the references are provided in Table 4.

Table 4. Proven Practices with Physical Architecture Development. (SEBoK Original)

Practice	Description
Modularity	Restrict the number of interactions between the system elements and consider the modularity principle (maximum of consistency inside the system element, minimum of physical interfaces with outside) as the right way for architecting systems.
Focus on Interfaces	Focusing on interfaces rather than on system elements is another key element of a successful architecture and design for abstract levels of systems.

References

Works Cited

ISO/IEC. 2007. Systems Engineering – Application and Management of The Systems Engineering Process. Geneva, Switzerland: International Organization for Standards (ISO)/International Electrotechnical Commission (IEC), ISO/IEC 26702:2007.

ISO/IEC/IEEE. 2015. *Systems and Software Engineering -- System Life Cycle Processes*. Geneva, Switzerland: International Organisation for Standardisation / International Electrotechnical Commissions / Institute of Electrical and Electronics Engineers. ISO/IEC/IEEE 15288:2015.

OMG. 2010. *OMG Systems Modeling Language specification*, version 1.2, July 2010. http://www.omg.org/technology/documents/spec_catalog.htm.

Faisandier, A. 2012. *Systems Architecture and Design*. Belberaud, France: Sinergy'Com.

Primary References

ANSI/IEEE. 2000. *Recommended practice for architectural description for software-intensive systems*. New York, NY: American National Standards Institute (ANSI)/Institute of Electrical and Electronics Engineers (IEEE), ANSI/IEEE 1471-2000.

INCOSE. 2015. *Systems Engineering Handbook* - A Guide for System Life Cycle Processes and Activities'', version 4.0. Hoboken, NJ, USA: John Wiley and Sons, Inc, ISBN: 978-1-118-99940-0

ISO/IEC/IEEE. 2015. *Systems and Software Engineering -- System Life Cycle Processes*. Geneva, Switzerland: International Organisation for Standardisation / International Electrotechnical Commissions / Institute of Electrical and Electronics Engineers. ISO/IEC/IEEE 15288:2015.

ISO/IEC/IEEE. 2011. *Systems and Software Engineering - Architecture Description*. Geneva, Switzerland: International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC)/Institute of Electrical and Electronics Engineers (IEEE), ISO/IEC/IEEE 42010.

Additional References

Maier, M., and E. Rechtin. 2009. *The Art of Systems Architecting,* 3rd ed. Boca Raton, FL, USA: CRC Press.

Holland, J.H. 2006. "Studying Complex Adaptive Systems." *Journal of Systems Science and Complexity*.19(1): 1-8. http://hdl.handle.net/2027.42/41486

Thome, B. 1993. *Systems Engineering, Principles & Practice of Computer-Based Systems Engineering.* New York, NY, USA: Wiley.

The Open Group. 2011. *TOGAF*, version 9.1. Hogeweg, The Netherlands: Van Haren Publishing. Accessed August 29, 2012. Available at: https://www2.opengroup.org/ogsys/jsp/publications/PublicationDetails.jsp?catalogno=g116.

Zachman, J. 2008. "John Zachman's Concise Definition of The Zachman Framework™." Zachman International Enterprise Architecture. Accessed August 29, 2012. Available at: http://www.zachman.com/about-the-zachman-framework.

< Previous Article | Parent Article | Next Article >

SEBoK v. 2.0, released 1 June 2019