1

SYSTEMS ENGINEERING INTRODUCTION

1.1 WHAT IS SYSTEMS ENGINEERING?

Systems Engineering (SE)

Our world and the systems we engineer continue to become more complex and interrelated. SE is an integrative approach to help teams collaborate to understand and manage systems and their complexity and deliver successful systems. The SE perspective is based on systems thinking—a perspective that sharpens our awareness of wholes and how the parts within those wholes interrelate (incose.org, *About Systems Engineering*). SE aims to ensure the pieces work together to achieve the objectives of the whole. SE practitioners work within a project team and take a holistic, balanced, life cycle approach to support the successful completion of system projects (INCOSE Vision 2035, 2022). SE has the responsibility to realize systems that are *fit for purpose*, namely that systems accomplish their intended purposes and be resilient to effects in real-world operation, while minimizing unintended actions, side effects, and consequences (Griffin, 2010).

Definition of SE

INCOSE Definitions (2019) and ISO/IEC/IEEE 15288 (2023) define:

> **Systems Engineering** is a transdisciplinary and integrative approach to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods.

INCOSE Definitions (2019) elaborates:
SE focuses on:

INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Fifth Edition.
Edited by David D. Walden, Thomas M. Shortell, Garry J. Roedler, Bernardo A. Delicado, Odile Mornas, Yip Yew-Seng, and David Endler.

- establishing, balancing and integrating stakeholders' goals, purpose and success criteria, and defining actual or anticipated stakeholder needs, operational concepts, and required functionality, starting early in the development cycle;
- establishing an appropriate life cycle model, process approach and governance structures, considering the levels of complexity, uncertainty, change, and variety;
- generating and evaluating alternative solution concepts and architectures;
- baselining and modeling requirements and selected solution architecture for each stage of the endeavor;
- performing design synthesis and system verification and validation;
- while considering both the problem and solution domains, taking into account necessary enabling systems and services, identifying the role that the parts and the relationships between the parts play with respect to the overall behavior and performance of the system, and determining how to balance all of these factors to achieve a satisfactory outcome.

SE provides facilitation, guidance, and leadership to integrate the relevant disciplines and specialty groups into a cohesive effort, forming an appropriately structured development process that proceeds from concept to development, production, utilization, support, and eventual retirement.

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

The goal of all SE activities is to manage risk, including the risk of not delivering what the acquirer wants and needs, the risk of late delivery, the risk of excess cost, and the risk of negative unintended consequences. One measure of utility of SE activities is the degree to which such risk is reduced. Conversely, a measure of acceptability of absence of a SE activity is the level of excess risk incurred as a result.

Definitions of System

While the concepts of a *system* can generally be traced back to early Western philosophy and later to science, the concept most familiar to SE practitioners is often traced to Ludwig von Bertalanffy (1950, 1968) in which a system is regarded as a "whole" consisting of interacting "parts."

INCOSE Definitions (2019) and ISO/IEC/IEEE 15288 (2023) define:

> A **system** is an arrangement of parts or elements that together exhibit behavior or meaning that the individual constituents do not.

A system is sometimes considered as a product or as the services it provides.

In practice, the interpretation of its meaning is frequently clarified using an associative noun (e.g., medical system, aircraft system). Alternatively, the word "system" is substituted simply by a context-dependent synonym (e.g., pacemaker, aircraft), though this potentially obscures a system principles perspective.

A complete system includes all of the associated equipment, facilities, material, computer programs, firmware, technical documentation, services, and personnel required for operations and support to the degree necessary for self-sufficient use in its intended environment.

INCOSE Definitions (2019) elaborates:
Systems can be either physical or conceptual, or a combination of both. Systems in the physical universe are composed of matter and energy, may embody information encoded in matter-energy carriers, and exhibit observable behavior. Conceptual systems are abstract systems of pure information, and do not directly exhibit behavior, but exhibit "meaning." In both cases, the system's properties (as a whole) result, or emerge, from:

a) the parts or elements and their individual properties,
b) the relationships and interactions between and among the parts, the system, other external systems (including humans), and the environment.

SE practitioners are especially interested in systems which have or will be "systems engineered" for a purpose. Therefore, INCOSE Definitions (2019) defines:

> An **engineered system** is a system designed or adapted to interact with an anticipated operational environment to achieve one or more intended purposes while complying with applicable constraints.

"Engineered systems" may be composed of any or all of the following elements: people, products, services, information, processes, and/or natural elements.

Origins and Evolution of SE

Aspects of SE have been applied to technical endeavors throughout history. However, SE has only been formalized as an engineering discipline beginning in the early to middle of the twentieth century (INCOSE Vision 2035, 2022). The term "systems engineering" dates to Bell Telephone Laboratories in the early 1940s (Fagen, 1978; Hall, 1962; Schlager, 1956). Fagen (1978) traces the concepts of SE within the Bell System back to early 1900s and describes major applications of SE during World War II. The British used multidisciplinary teams to analyze their air defense system in the 1930s (Martin, 1996). The RAND Corporation was founded in 1946 by the United States Air Force and claims to have created "systems analysis." Hall (1962) asserts that the first attempt to teach SE as we know it today came in 1950 at MIT by Mr. Gilman, Director of Systems Engineering at Bell. TRW (now a part of Northrop Grumman) claims to have "invented" SE in the late 1950s to support work with ballistic missiles. Goode and Machol (1957) authored the first book on SE in 1957. In 1990, a professional society for SE, the National Council on Systems Engineering (NCOSE), was founded by representatives from several US corporations and organizations. As a result of growing involvement from SE practitioners outside of the US, the name of the organization was changed to the International Council on Systems Engineering (INCOSE) in 1995 (incose.org, *History of Systems Engineering*; Buede and Miller, 2016).

With the introduction of the international standard ISO/IEC 15288 in 2002, the discipline of SE was formally recognized as a preferred mechanism to establish agreement for the creation of products and services to be traded between two or more organizations—the supplier(s) and the acquirer(s). This handbook builds upon the concepts in the latest edition of ISO/IEC/IEEE 15288 (2023) by providing additional context, definitions, and practical applications. Table 1.1 provides a list of key SE standards and guides related to the content of this handbook.

TABLE 1.1 SE standards and guides

Reference	Title
ISO/IEC/IEEE 15026	Systems and software engineering—Systems and software assurance (Multi-part standard)
ISO/IEC/IEEE 15288	Systems and software engineering—System life cycle processes
IEEE/ISO/IEC 15289	Systems and software engineering—Content of life cycle information items (documentation)
ISO/IEC/IEEE 15939	Systems and software engineering—Measurement process
ISO/IEC/IEEE 16085	Systems and software engineering—Life cycle processes—Risk management
ISO/IEC/IEEE 16326	Systems and software engineering—Life cycle processes—Project management
ISO/IEC/IEEE 21839	Systems and software engineering—System of systems (SoS) considerations in life cycle stages of a system
ISO/IEC/IEEE 21840	Systems and software engineering—Guidelines for the utilization of ISO/IEC/IEEE 15288 in the context of system of systems (SoS)
ISO/IEC/IEEE 21841	Systems and software engineering—Taxonomy of systems of systems
ISO/IEC/IEEE 24641	Systems and software engineering—Methods and tools for model-based systems and software engineering

(Continued)

TABLE 1.1 (Continued)

Reference	Title
ISO/IEC/IEEE 24748–1	Systems and software engineering—Life cycle management—Part 1: Guidelines for life cycle management
ISO/IEC/IEEE 24748–2	Systems and software engineering—Life cycle management—Part 2: Guidelines for the application of ISO/IEC/IEEE 15288
ISO/IEC/IEEE 24748–4	Systems and software engineering—Life cycle management—Part 4: Systems engineering planning
ISO/IEC/IEEE 24748–6	Systems and software engineering—Life cycle management—Part 6: System integration engineering
ISO/IEC/IEEE 24748–7	Systems and software engineering—Life cycle management—Part 7: Application of systems engineering on defense programs
ISO/IEC/IEEE 24748–8 / IEEE 15288.2	Systems and software engineering—Life cycle management—Part 8: Technical reviews and audits on defense programs
ISO/IEC/IEEE 24765	Systems and software engineering—Vocabulary
ISO/IEC/IEEE 26550	Software and systems engineering—Reference model for product line engineering and management
ISO/IEC/IEEE 26580	Software and systems engineering—Methods and tools for the feature-based approach to software and systems product line engineering
ISO/IEC/IEEE 29148	Systems and software engineering—Life cycle processes—Requirements engineering
ISO/IEC/IEEE 42010	Systems and software engineering—Architecture description
ISO/IEC/IEEE 42020	Software, systems and enterprise—Architecture processes
ISO/IEC/IEEE 42030	Software, systems and enterprise—Architecture evaluation framework
ISO/IEC 29110	Systems and Software Engineering Standards and Guides for Very Small Entities (VSEs) (Multi-part set)
ISO/IEC 31000	Risk management
ISO/IEC 31010	Risk management—Risk assessment techniques
ISO/IEC 33060	Process assessment—Process assessment model for system life cycle processes
ISO/PAS 19450	Automation systems and integration—Object-Process Methodology (OPM)
ISO 10007	Quality management—Guidelines for configuration management
ISO 10303-233	Industrial automation systems and integration—Product data representation and exchange—Part 233: Application protocol: Systems engineering
NIST SP 800–160 Vol. 1	Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems
NIST SP 800–160 Vol. 2	Developing Cyber-Resilient Systems: A Systems Security Engineering Approach
OMG SysML™	OMG Systems Modeling Language
SEBoK	Guide to the Systems Engineering Body of Knowledge (SEBoK)
SAE-EIA 649C	Configuration Management Standard
SAE 1001	Integrated Project Processes for Engineering a System (Note: Replaced ANSI/EIA 632)
ANSI/AIA.A G.043B	Guide to the Preparation of Operational Concept Documents
CMMI	CMMI® V2.0

1.2 WHY IS SYSTEMS ENGINEERING IMPORTANT?

The purpose of SE is to conceive, develop, produce, utilize, support, and retire the right product or service within budget and schedule constraints. Delivering the right product or service requires a common understanding of the current system state and a common vision of the system's future states, as well as a methodology to transform a set of stakeholder needs, expectations, and constraints into a solution. The right product or service is one that accomplishes the required service or mission. A common vision and understanding, shared by acquirers and suppliers, is achieved through application of proven methods that are based on standard approaches across people, processes, and tools. The application of these methods is continuous throughout the system's life cycle.

SE is particularly important in the presence of complexity (see Section 1.3.7). Most current systems are formed by integrating commercially available products or by integrating independently managed and operated systems to provide emergent capabilities which increase the level of complexity (see Sections 4.3.3 and 4.3.6). This increased reliance on off-the-shelf and systems of systems has significantly reduced the time from concept definition to market availability of products. Over the years between 1880 and 2000, average 25% market penetration has been reduced by more than a factor of four as illustrated in Figure 1.1.

In response to complexity and compressed timelines, SE methods and tools have become more adaptable and efficient. Introduction of agile methods (see Section 4.2.2) and SE modeling language standards such the Systems Modeling Language (SysML) have allowed SE practitioners to manage complexity and increase the implementation of a common system vision (see bottom of Figure 1.1). Model Based SE (MBSE) methods adoption continues to grow (see Section 4.2.1), particularly in the early conceptual design and requirements analysis (SEBOK, *Emerging Topics*). MBSE research literature continues to report on the increased productivity and quality of design and promises further progression toward a digital engineering (DE) approach, where data is transparent and cooperation optimized across all engineering disciplines. Standards organizations are updating or developing new approaches that take DE into consideration. SE will have to address this new digital representation of the system as DE becomes the way of doing business (see Section 5.4). The rapid evolution and introduction of Artificial Intelligence (AI) and Machine Learning (ML) into SE further increases complexity of verifiability, safety, and trust of self-learning and evolving systems.

The overall value of SE has been the subject of studies and papers from many organizations since the introduction of SE. A 2013 study was completed at the University of South Australia to quantify the return on investment (ROI) of SE activities on overall project cost and schedule (Honour, 2013). Figure 1.2 compares the total SE effort with cost

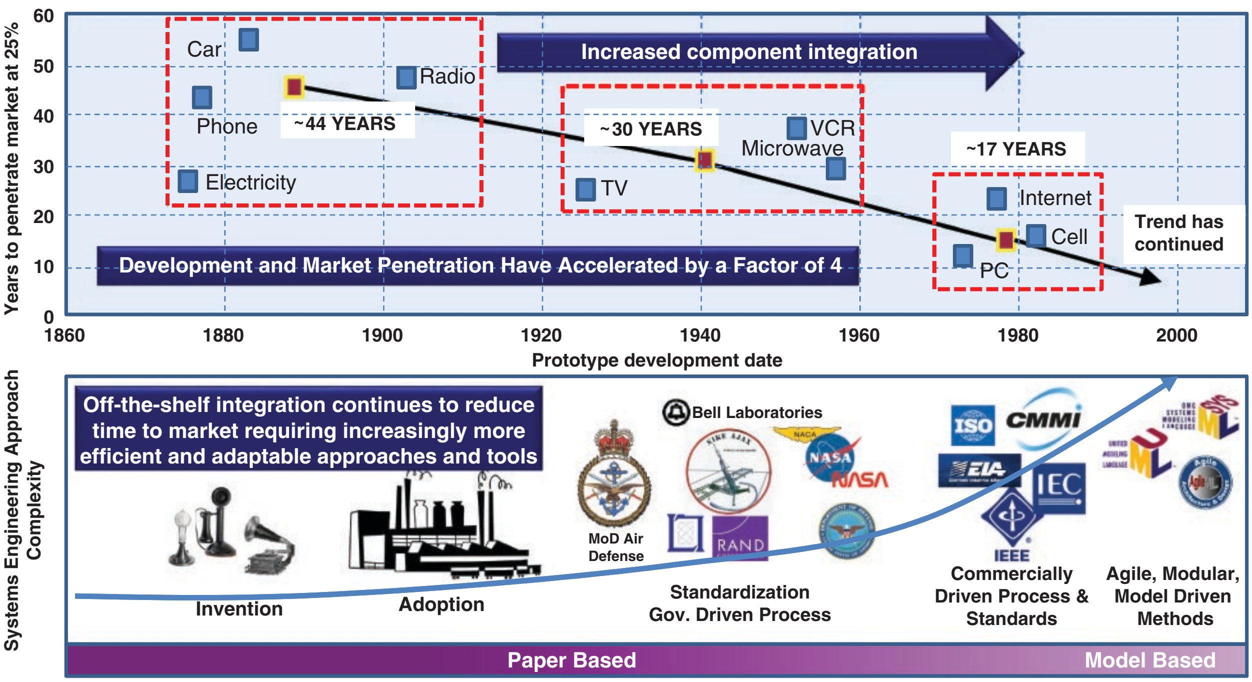

FIGURE 1.1 Acceleration of design to market life cycle has prompted development of more automated design methods and tools. INCOSE SEH original figure created by Amenabar. Usage per the INCOSE Notices page. All rights reserved.

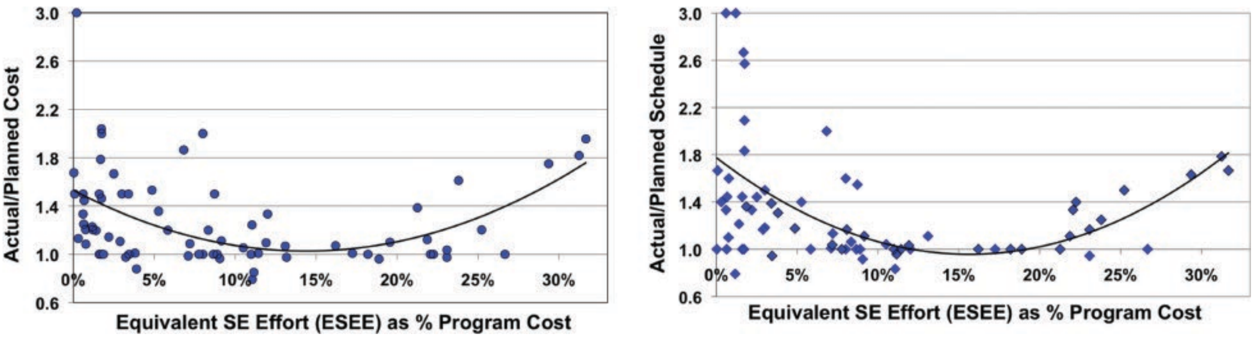

FIGURE 1.2 Cost and schedule overruns correlated with SE effort. From Honour (2013) with permission from University of South Wales. All other rights reserved.

compliance (left figure) and schedule performance (right figure). In both graphs, increasing the percentage of SE within the project results in better success up to an optimum level, above which SE ROI is diminished above those total program expenditure levels due to increased unwarranted processes. Study data shows that SE effort had a significant, quantifiable effect on project success, with correlation factors as high as 80%. Results show that the optimum level of SE effort for a normalized range of 10% to 14% of the total project cost.

The ROI of adding additional SE activities to a project is shown in Table 1.2, and it varies depending on the level of SE activities already in place. If the project is using no SE activities, then adding SE carries a 7:1 ROI; for each cost unit of additional SE, the project total cost will reduce by 7 cost units. At the median level of the projects interviewed, additional SE effort carries a 3.5:1 ROI.

A joint 2012 study by the National Defense Industrial Association (NDIA), the Institute of Electrical and Electronic Engineers (IEEE), and the Software Engineering Institute (SEI) of Carnegie Mellon University (CMU) surveyed 148 development projects and found clear and significant relationships between the application of SE activities and the performance of those projects as seen in Figure 1.3 (Elm and Goldenson, 2012). The study broke the projects by the maturity of their SE processes as measured by the quantity and quality of specific SE work products and considered the complexity of each project and the maturity of the technologies being implemented (n=number of projects). It also assessed the levels of project performance, as measured by satisfaction of budget, schedule, and technical requirements. The left column represents those projects deploying lower levels of SE expertise and capability. Among these projects, only 15% delivered higher levels of project performance and 52% delivered lower levels of project performance. The center column represents those projects deploying moderate levels of SE expertise and capability. Among these projects, the number delivering higher levels of project performance increased to 24% and those delivering lower levels decreased to 29%. The right column represents those projects deploying higher levels of SE expertise and capability. For these projects, the number delivering higher levels of project performance increased substantially

TABLE 1.2 SE return on investment

Current SE effort (% of program cost)	Average cost overrun (%)	ROI for additional SE effort (cost reduction $ per $ SE added)
0	53	7.0
5	24	4.6
7.2 (median of all programs)	15	3.5
10	7	2.1
15	3	–0.3
20	10	–2.8

From Honour (2013) with permission from University of South Wales. All other rights reserved.

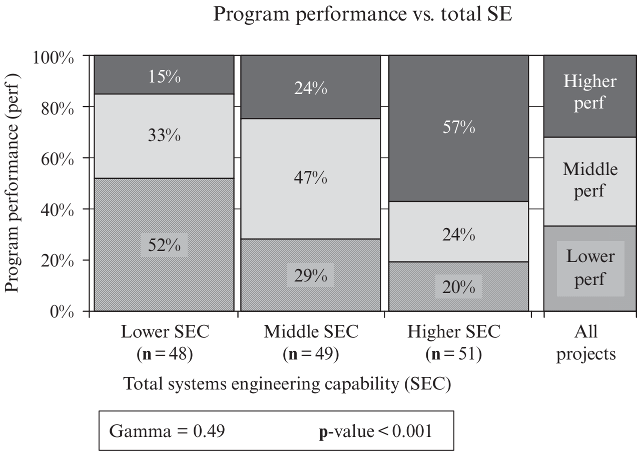

FIGURE 1.3 Project performance versus SE capability. From Elm and Goldenson (2012) with permission from Carnegie Mellon University. All other rights reserved.

to 57%, while those delivering lower levels decreased to 20%. As Figure 1.3 shows, well-applied SE increases the probability of successfully developing an engineered system.

A 1993 Defense Acquisition University (DAU) statistical analysis on US Department of Defense (DoD) projects examined spent and committed life cycle cost (LCC) over time (DAU, 1993). As illustrated notionally in Figure 1.4, an important result from this study is that by the time approximately 20% of the actual costs have been accrued, over 80% of the total LCC has already typically been committed. Figure 1.4 also shows that it is less costly to fix or address issues if they are identified early. Good SE practice is the means by which the issues are identified and ensures that the understanding obtained is applied as appropriate during the life cycle, thus reducing technical debt.

INCOSE maintains value proposition statements (INCOSE Value Strategic Initiative Report, 2021) as tailored to different areas and industries. Areas covered include individual INCOSE membership, organizational INCOSE membership, INCOSE SE certification, and the discipline of SE. Industries include commercial, government, and nonprofit organizations. A sample of these findings includes:

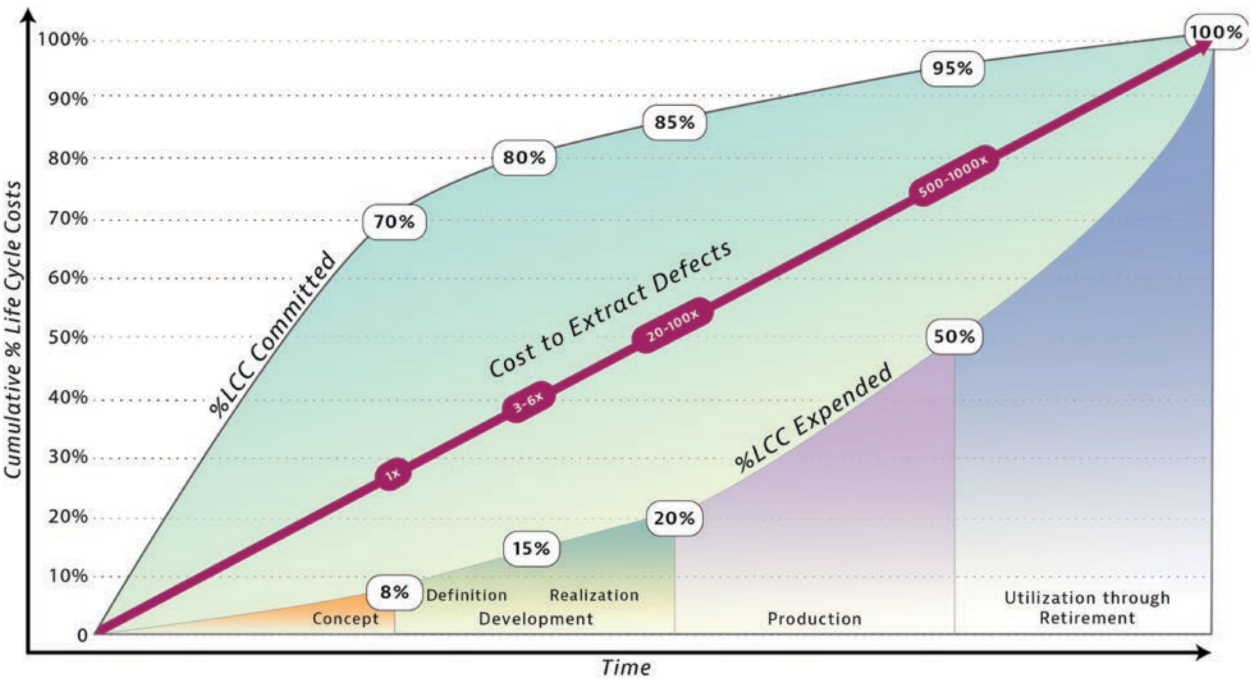

FIGURE 1.4 Life cycle costs and defect costs against time. INCOSE SEH original figure created by Walden derived from DAU (1993). Usage per the INCOSE Notices page. All other rights reserved.

- *Value of SE to the Commercial/Market-Driven Industry*: Companies and other enterprises in commercial industry will benefit from the internal practice of professional SE by having enhanced their capability for the development of innovative products and services for distribution in both mature and immature markets, in a more efficient and competitive manner.
- *Value of SE to Government/Infrastructure/Aerospace/Defense Industry*: SE provides a tailorable, systematic approach to all stages of a project, from concept to retirement. SE can accommodate different approaches including agile and sequential and facilitate commonality and open architectures to ensure lower acquisition, maintenance, and upgrade costs. By confirming correct and complete requirements and requirements allocations, the resulting design has fewer and less significant changes resulting in improved overall cost and schedule performance.
- *Value of SE to Nonprofit/Research Industry*: A nonprofit enterprise will benefit from the internal practice of professional SE by having enhanced their capability for the development of innovative client services in a more efficient and effective manner. An enterprise engaged in basic or applied research will benefit from the internal practice of SE by having enhanced its capabilities for discovery and invention that supports technology development in a more effective manner.

1.3 SYSTEMS CONCEPTS

Important system concepts include the system of interest (SoI), the system environment, and external systems. The boundaries between the system and the surrounding elements are important to understand. These boundaries separate the SoI, enabling systems, interoperating systems, and interfacing systems, supporting the SE practitioner in properly accounting for all the necessary elements which comprise the whole system context. Part of the system concept are the system's modes and states which are fundamental system behavior characteristics important to SE. Systems can be hierarchical in their structural organization, or they can be complex where hierarchy is not always present. The system concepts encompass all types of systems structures and support the SE practitioner with a framework in which to engineer a system.

1.3.1 System Boundary and the System of Interest (SoI)

General System Concepts An external view of a system must introduce elements that specifically do not belong to the system but do interact with the system. This collection of elements is called the *system environment or context* and can include the users (or operators) of the system. It is important to understand that the system environment or context is not limited to the operating environment, but also includes external systems that interface with or support the system at any time of the life cycle.

The internal and external views of a system give rise to the concept of a *system boundary*. In practice, the system boundary is a "line of demarcation" between the system under consideration, called the system of interest (SoI), and its greater context. It defines what belongs to the system and what does not. The system boundary is not to be confused with the subset of elements that interact with the environment.

The *functionality* of a system is typically expressed in terms of the interactions of the system with its operating environment, especially the users. When a system is considered as an integrated combination of interacting elements, the functionality of the system derives not just from the interactions of individual elements with the environmental elements but also from how these interactions are influenced by the organization (interrelations) of the system elements. This leads to the concept of *system architecture*, which ISO/IEC/IEEE 42020 (2019) defines as:

> Fundamental concepts or properties of an entity in its environment and governing principles for the realization and evolution of this entity and its related life cycle processes.

This definition speaks to both the internal and external views of the system and shares the concepts from the definitions of a system (see Section 1.1).

Scientific Terminology Related to System Concepts In general, *engineering* can be regarded as the practice of creating and sustaining systems, services, devices, machines, structures, processes, and products to improve the quality of life—getting things done effectively and efficiently. The repeatability of experiments demanded by science is critical for delivering practical engineering solutions that have commercial value. Engineering in general, and SE in particular, draw heavily from the terminology and concepts of science.

An *attribute* of a system (or system element) is an observable characteristic or property of the system (or system element). For example, among the various attributes of an aircraft is its air speed. Attributes are represented symbolically by variables. Specifically, a *variable* is a symbol or name that identifies an attribute. Every variable has a domain, which could be but is not necessarily measurable. A *measurement* is the outcome of a process in which the SoI interacts with an observation system under specified conditions. The outcome of a measurement is the assignment of a *value* to a variable. A system is in a *state* when the values assigned to its attributes remain constant or steady for a meaningful period of time (Kaposi and Myers, 2001). In SE and software engineering, the *system elements* (e.g., software objects) have *processes* (e.g., operations) in addition to attributes. These have the binary logical values of being either *idle* or *executing*. A complete description of a system state therefore requires values to be assigned to both attributes and processes. *Dynamic behavior* of a system is the time evolution of the system state. *Emergent behavior* is a behavior of the system that cannot be understood exclusively in terms of the behavior of the individual system elements. See Section 1.3.2 for further information on emergent behavior and Section 1.3.6 for more information on states and modes.

The key concept used for problem solving is the *black box/white box* (also known as *opaque box/transparent box*) system representation. The *black box (opaque box)* representation is based on an external view of the system (attributes). The *white box (transparent box)* representation is based on an internal view of the system (attributes and structure of the elements). Both representations are useful to the SE practitioner and there must be an understanding of the relationship between the two. A system, then, is represented by the external attributes of the system, its internal attributes and structure, and the interrelationships between these that are governed by the laws of science.

1.3.2 Emergence

Emergence describes the phenomenon that whole entities exhibit properties which are meaningful only when attributed to the whole, not to its elements. Every model of human activity system exhibits properties as a whole entity that derive from its element activities and their structure, but cannot be reduced to them (Checkland, 1999). Emergence is a fundamental property of all systems (Sillitto and Dori, 2017). According to Rousseau et al. (2018), emergence derives from the systems science concept of "properties the system has but the elements by themselves do not."

System elements interact between themselves and can create desirable or undesirable phenomena called *emergent properties* such as inhibition, interference, resonance, or reinforcement of any property. Emergent properties can also result from the interaction between the system and its environment. Many engineering disciplines include emergence as a property. For example, system safety (Leveson, 1995) and resilience (Rasoulkahni, 2018) are examples of emergent properties of engineered systems (see Sections 3.1.11 and 3.1.9, respectively).

Definition of the architecture of the system includes an analysis of interactions between system elements in order to reinforce desirable and prevent undesirable emergent properties. According to Rousseau et al. (2019), the systemic virtue of emergent properties are used during systems architecture and design definition to highlight necessary derived functions and internal physical or environmental constraints (see Sections 2.3.5.4 and 2.3.5.5, respectively). Corresponding derived requirements should be added to system requirements baseline when they impact the SoI.

Calvo-Amodio and Rousseau (2019) explain how emergence applies to systems in which complexity is dominant. Complexity dominance, they say, encourages us to consider the significance of the difference between kinds of

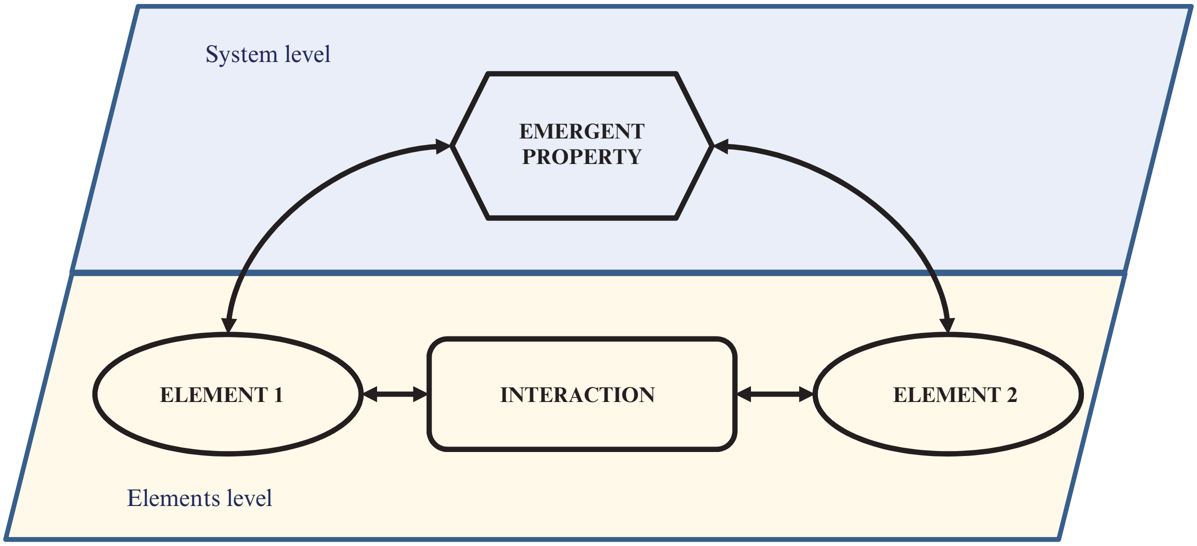

FIGURE 1.5 Emergence. INCOSE SEH original figure created by Jackson. Usage per the INCOSE Notices page. All other rights reserved.

complexity and degrees of complexity systems have. Doing so enables the SE practitioner to use variety engineering to manage complexity accordingly.

Figure 1.5 illustrates how the interaction between elements can result in emergent properties in any kind of system. This figure illustrates the basic rules of emergence. First, individual elements cannot exhibit higher-level system emergence. Second, two or more elements are required for emergence. Finally, emergence occurs at a level above the individual elements.

1.3.3 Interfacing Systems, Interoperating Systems, and Enabling Systems

External systems are systems beyond (or outside of) the SoI boundary. *Interfacing systems* are external systems that share an interface (e.g., physical, material, energy, data/information) with the SoI. Typically, humans also interface with the SoI throughout the SoI's life cycle stages. *Interoperating systems* are interfacing systems that interface with the SoI in its operational environment to perform a common function that supports the SoI's primary purpose. The set of SoI and interoperating systems can be seen as a system of systems (see Section 4.3.6). *Enabling systems* are external systems that facilitate the life cycle activities of the SoI but are not a direct element of the operational environment. The enabling systems provide services that are needed by the SoI during one or more life cycle stages. Some enabling systems share an interface with the SoI and some do not. Examples of enabling systems include collaboration development systems, production systems, and logistics support systems. Table 1.3 gives examples of these types of external systems.

During the life cycle stages for an SoI, it is necessary to concurrently consider interfacing, interoperating, and enabling systems along with the SoI. Otherwise, important requirements may not be identified, which will lead to significant costs in the further course of system development. Typical pitfalls include assuming that a new enabling system will come online in time to support the development of the SoI or that an existing enabling system will be available for the duration of the life cycle of the SoI. A delay in an enabling system coming online or the loss of an existing enabling system can lead to significant issues with the development and deployment of the SoI. In addition, horizontal and vertical integration considerations (see Section 2.3.5.8) may arise from the system context represented by interfacing, interoperating, and enabling systems.

TABLE 1.3 Examples for systems interacting with the SoI

SOI and External Systems	Interfacing System	Interoperating System	Enabling System
Aircraft			
Flight simulator	No	No	Yes
Fuel Truck	Yes	No	Yes
Remote Maintenance	Yes	Yes	Yes
Communication system	Yes	Yes	No
Runway	Yes	No	No
Automobile			
SE Tool	No	No	Yes
Car carrier	Yes	No	Yes
Diagnosis system	Yes	Yes	Yes
Parking assistant	Yes	Yes	No
Windshield snow cover	Yes	No	No

1.3.4 System Innovation Ecosystem

Sections 1.3.1 and 1.3.3 describe the system boundary and external systems in the overall context of the SoI. This section focuses on learning. Over single, and eventually multiple life cycles, engineered system innovation may be viewed as a form of group learning by "ecosystems" composed of individuals, teams, enterprises, supply chains, markets, and societies. Effective innovation requires effective learning and adaptation at a group level across these ecosystems and brings related challenges. To represent, plan, analyze, and improve such performance, the neutral descriptive System Innovation Ecosystem Pattern has been found to be useful (Schindel and Dove, 2016) (Schindel 2022b). Figure 1.6 provides a high-level view of that multiple-layered descriptive model, further discussed as a formal pattern in Section 3.2.6.

Figure 1.6 identifies three top-level system boundaries:

1. **System 1 – The Engineered System** may be a product developed for a market, a defense system created under contract, a service-providing system, or other system subject to SE life cycle management. It is shown in its larger environment, the Life Cycle Project Management System (System 2). System 1 examples include Medical Devices, Aircraft, Consumer Packaged Goods, and Gas Turbine Engines. This system is typically referred to as the engineered SoI in this handbook.
2. **System 2 – The Life Cycle Project Management System** provides the environment of System 1 over its life cycle, including the life cycle management processes responsible for System 1—described in Part II. System 2, a socio-technical system of people, processes, and facilities, is responsible to learn about System 1 and its environment, and to effectively apply that learning in the life cycle management by System 2. System 2 examples include System Requirements Definition Processes, Verification Processes, Product Manufacturing Processes, Product Distribution Processes, Product Sustainment Systems, Product Life Cycle Management (PLM) Information Systems, and Product Digital Twin Systems.
3. **System 3 – The Enterprise Process and Innovation System** contains System 2 and is responsible for learning about and improving System 2. In that sense, System 3 includes formal life cycle management for the processes of System 2. System 3 contains the "organizational change management" for advancing and adapting System 2 as a recognized formal system in its own right. System 3 examples include Product Life Cycle Management Processes, Program and Project Configuration and Tailoring Processes, Engineering Recruitment, Education, and Advancement Processes, Product Development Methodology Descriptions, Engineering Automation Tooling Acquisition and Development, Development Process Performance Analysis Systems, Regulatory Authorities, Engineering Professional Societies, and Engineering Facilities Construction and Acquisition.

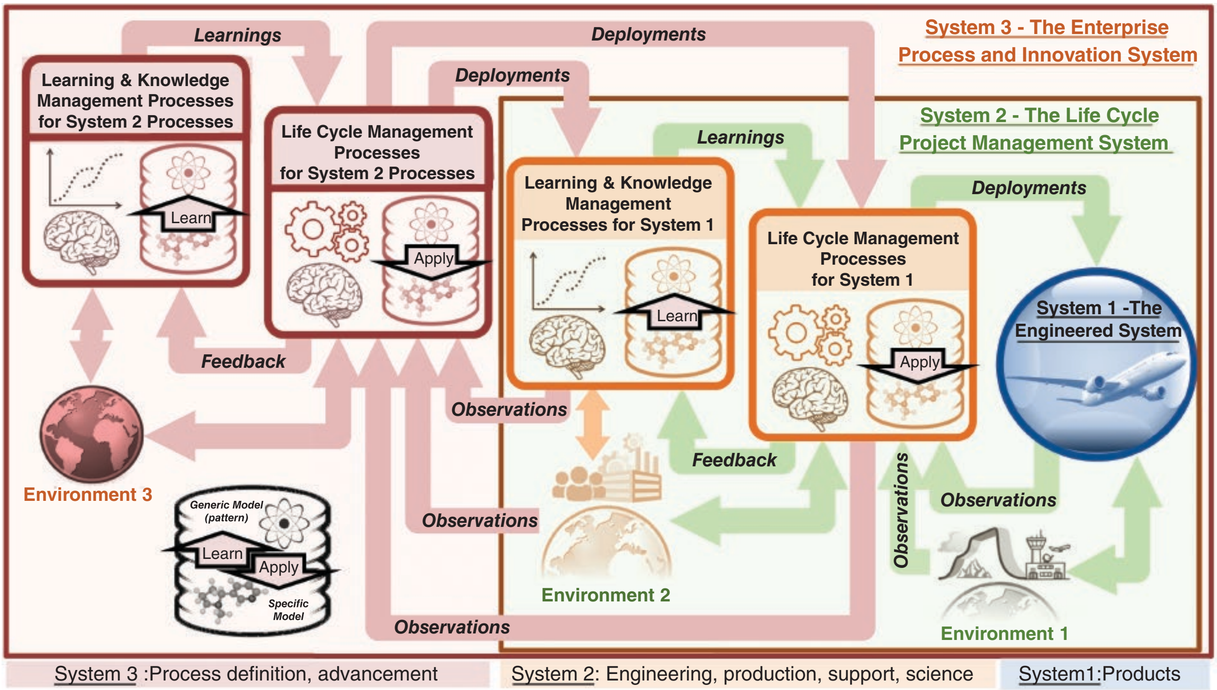

FIGURE 1.6 System innovation ecosystem pattern. From Schindel and Dove (2016) and Schindel (2022b). Used with permission. All other rights reserved.

The System Innovation Ecosystem Pattern emphasizes the learning and execution aspects of the enterprise ecosystem and directly integrates the SE life cycle processes described in Part II of this Handbook. Those processes are applied to two different managed SoIs (System 1 and System 2) and explicate the processes of learning versus application in each of the SE life cycle processes, along with how, and how effectively, execution is coupled with prior learning. The (configurable) System Innovation Ecosystem Pattern intentionally describes any engineering environment, whether effective in its learning and adaptation or not. It is intended as a descriptive, not prescriptive, reference model that can be used to plan and analyze any engineering and life cycle management ecosystem. So, while the "learned models" shown inside System 2 describe knowledge of System 1 (The Engineered System), the models shown inside System 3 describe knowledge of System 2 (The Life Cycle Project Management System).

The formal System Innovation Ecosystem Pattern includes the ability to be configured specific to a local enterprise, project, or supply chain, and for use to plan a series of migration increments representing advancing System 2 capabilities. For more details, refer to Section 3.2.6 and the INCOSE S*Patterns Primer (2022).

1.3.5 The Hierarchy within a System

As explained in Section 1.1, "A system is an arrangement of parts or elements." A *system element* is a member of a set of elements that constitute a system (ISO/IEC/IEEE 15288, 2023). A system element is a discrete part of a system that can be implemented to fulfil specified requirements. Hardware, software, data, humans, processes (e.g., processes for providing service to users), procedures (e.g., operator instructions), facilities, materials, and naturally occurring entities or any combination are examples of system elements.

In the ISO/IEC/IEEE 15288 (2023) usage of terminology, the system elements can be atomic (i.e., not further decomposed), or they can be systems on their own merit (i.e., decomposed into further subordinate system elements).

A system element that needs only a black box (also known as opaque box) representation (i.e., external view) to capture its requirements and confidently specify its real-world solution definition can be regarded as atomic. Decisions to make, buy, or reuse the element can be made with confidence without further specification of the element.

One of the challenges of system definition is to understand what level of detail is necessary to define each system element and the interrelations between elements. The integration of the system elements must establish the relationship between the effects that organizing the elements has on their interactions and how these effects enable the system to achieve its purpose. One approach to defining the elements of a system and their interrelations is to identify a complete set of distinct system elements with regard only to their relation to the whole (system) by suppressing details of their interactions and interrelations. These considerations lead to the concept of *hierarchy* within a system. This is referred to as a *partitioning* of the system and the end result is called a *Product Breakdown Structure* (PBS) (see Section 2.3.4.1). As stated above, each element of the PBS can be either atomic or it can be at a higher level that could be viewed as a system itself. At any given level, the elements are grouped into distinct subsets of elements subordinated to a higher-level system, as illustrated in Figure 1.7. Thus, hierarchy within a system is an organizational representation of system structure using a partitioning relation.

The art of defining a hierarchy within a system relies on the ability of the SE practitioner to strike a balance between clearly and simply defining span of control and resolving the structure of the SoI into a complete set of system elements that can be implemented with confidence. Urwick (1956) suggested a possible heuristic for span of control, recommending that decomposition of any object in a hierarchy be limited to no more than seven subordinate elements, plus or minus two (7 +/–2). Others have also found this heuristic to be useful in other contexts (Miller, 1956). A level of design with too few subordinate elements is unlikely to have a distinct design activity. In this case, both design and verification activities may contain redundancy. In case of too many subordinate elements, it may be difficult to manage all the interfaces between the subordinate elements. In practice, the nomenclature and depth of the hierarchy can and should be adjusted to fit the nature of the system and the community of interest.

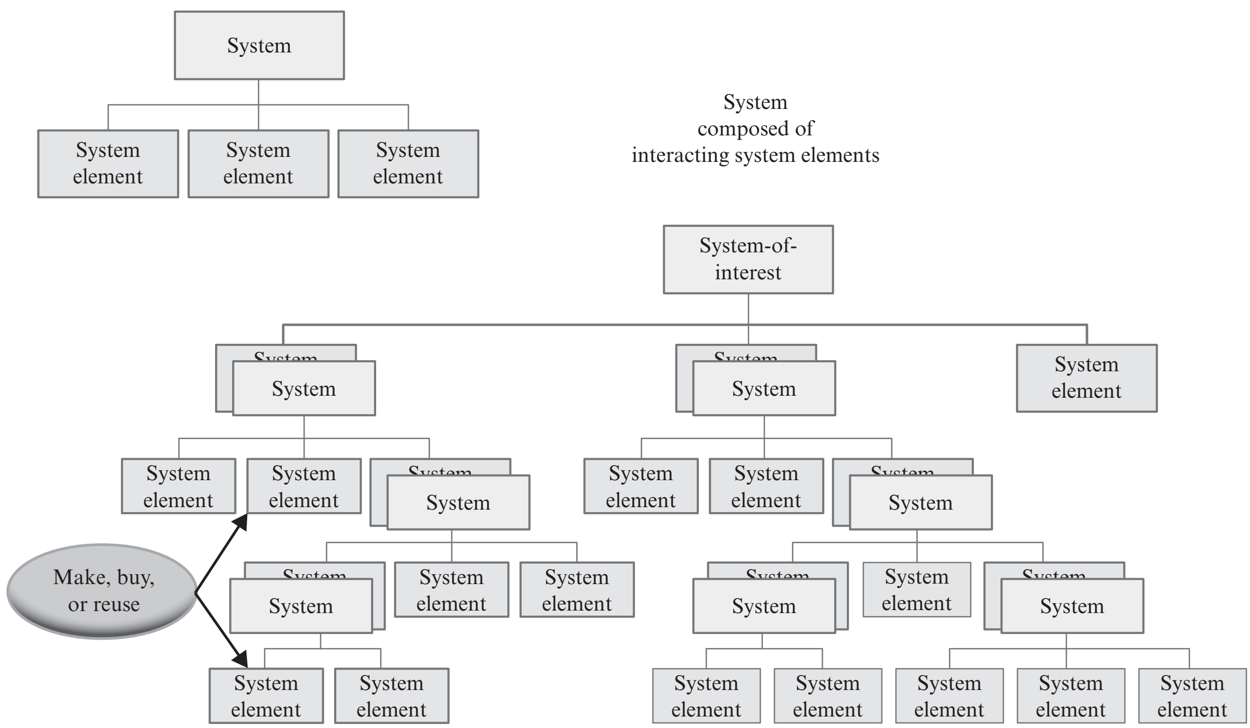

FIGURE 1.7 Hierarchy within a system. From ISO/IEC/IEEE 15288 (2023). Used with permission. All other rights reserved.

The interrelationships of system elements at a given architecture level of decomposition can be referred to as the horizontal view of the system. The horizontal view also includes requirements; integration, verification, or validation activities and results; various other related artifacts; and external elements. How the horizontal elements, activities, results, and artifacts are derived from or lead to higher-level systems and lower-level system element can be referred to as the vertical view of the system.

1.3.6 Systems States and Modes

States and modes are two related concepts that are used for defining and modeling system functional architectures and for modeling and managing system behaviors.

A *state* can be defined as:

> An observable and measurable … attribute used to characterize the current configuration, status, or performance-based condition of a System or Entity. (Wasson, 2016)

States are snapshots of a set of variables or measurements needed to describe fully the system's capabilities to perform the system's functions. *State variables* are the multidimensional list of variables that determine the state of the system. The list of variables does not change over time, but the values that these variables take do change over time (Buede and Miller, 2016). In control theory, the state of a dynamic system is a set of physical quantities, the specification of which (in Newtonian dynamics) completely determines the evolution of the system (Friedland, 2012). From the perspective of MBSE (see Section 4.2.1), "The state of the system is the most concise description of its past history."

The current system state and a sequence of subsequent inputs allow computation of the future states of the system. The state of a system contains all the information needed to calculate future responses without reference to the history of inputs and responses (Chapman, et al., 1992). Bonnet et al. (2017) states, "A state often directly reflects an operating condition or status on structural elements of the system (operational, failed, degraded, absent, etc.). States are also likely to represent the physical condition of a system element (full or empty fuel tank, charged or discharged battery, etc.). States can also be exploited to represent environment constraints (temperature, humidity, etc.)." If the system is transitioning from one state to another as time progresses, then time is one of the key attributes of the system. To monitor the system and manage it, the manager observes a state variable that is comprised of the appropriate collection of the system's attributes (Shafaat and Kenley, 2020).

A *mode* can be defined as:

> A distinct operating capability of the system during which some or all of the system's functions may be performed to a full or limited degree. (Buede and Miller, 2016)

For a personal computer, examples of modes are "off," "on," "waking up," "waiting," "reading from disk," "writing to disk," "computing," "printing," and, of course, "down" (Wymore, 1993). Modes are part of the system functional architecture and can be derived by affinity analysis of system use cases (Wasson, 2016). Various perspectives can be used to define the distinct operating modes of a system (Bonnet, et al., 2017), such as:

- the phases of mission operations (taxiing, taking-off, cruising, landing, etc.),
- the system operating conditions (connected, autonomous, etc.),
- the specific conditions in which the system is used (test, training, maintenance, etc.).

Transitioning from one mode to another is the result of decisions made by the system itself, its users, or external actors in order to adapt to new needs or new contexts (Bonnet, et al., 2017). Decisions that result in the system transitioning from one mode to another are typically based on the observed values of the state variables. When using models to depict system behavior, mode transitions are often based on triggering events that meet specified entry and exit criteria (Wasson, 2016).

1.3.7 Complexity

Systems engineering practitioners encounter a number of systems with simple, complicated, and complex characteristics. Many traditional systems engineering approaches and techniques work well for simple and complicated systems but do not handle complexity in systems (i.e., complex systems) well. Conversly, approaches and techniques that handle complexity well are also used in some complicated system contexts, especially when complex characteristics exist in some aspects of the system. Thus, care must be used to ensure the SE approaches and techniques for the SoI are appropriate and tailored for the type of system, especially with respect to its complexity. Complex systems are defined in the INCOSE publication “A Complexity Primer for Systems Engineers” (INCOSE Complexity Primer, 2021). A complex system has elements, the relationship between the states of which are weaved together so that they are not fully comprehended, leading to insufficient certainty between cause and effect. Complicated systems are less challenging. A complicated system has elements, the relationship between the states of which can be unfolded and comprehended, leading to sufficient certainty between cause and effect. Systems can also be simple. A simple system has elements, the relationship between the states of which, once observed, are readily comprehended. Complex systems can provide beneficial solutions yet also contain challenging characteristics. Complexity can result in positive behavior, such as self-organization and virtuous cycles of activity. However, intricate networks of evolving cause-and-effect relationships can lead to novel, nonlinear, and counterintuitive dynamics over time, resulting in suboptimal system operation, unintended consequences, and system obsolescence. The INCOSE Complexity Primer identifies 14 distinguishing characteristics that define complexity in a system. These characteristics provide insights into complexity, realizing that systems are not wholly complex: they are typically complex in some characteristics and complicated or even simple in others.

Traditional SE process for complicated systems takes a reductionist approach, whereby the problem is procedurally broken down into its parts (i.e., decomposition), solved, and reassembled to form the whole solution. This approach works well for complicated problems, where fixed, deterministic, or predictable patterns of behavior are required. However, these processes often do not perform well in complex environments, such as the challenges involved in designing autonomous vehicles or other socio-technical systems. A fundamentally different approach is required to understand the unexpected emergent interaction between the parts in the context of the whole through iterative exploration and adaptation (Snowden and Boone, 2007).

SE for complex systems requires a balance of linear, procedural methods for sorting through complicated and intricate tasks (e.g., systematic activity) and holistic, nonlinear, iterative methods for harnessing complexity (e.g., systems thinking). Complexity is not antithetical to simplicity, as even relatively simple systems can generate complex behavior. The INCOSE Complexity Primer provides guidance in the methods, approaches, and tools that may benefit complex systems engineering.

1.4 SYSTEMS ENGINEERING FOUNDATIONS

1.4.1 Uncertainty

There is uncertainty associated with much of the systems information and measurement data we use. This section provides a brief summary of the two major types of uncertainty, the sources of systems uncertainty, and decision making under uncertainty.

Types of Uncertainty. There are two types of uncertainties: epistemic and aleatory. In SE, *epistemic uncertainty* is due to our lack of knowledge about the potential demand for a new system and how a technology, system, or process will perform in the future, for example, the knowledge gap about key value attribute or about the acquirer’s preferences. *Aleatory uncertainty* is uncertainty due to randomness. If a technology, system, or process can perform a function, there will be always some inherent randomness in every performance measurement. Our system requirements process, and development decisions focus on reducing epistemic uncertainty (overcoming our lack of knowledge), but we can never completely reduce aleatory uncertainty in our development or operational measurement of system performance.

TABLE 1.4 Sources of system uncertainty

Sources of Uncertainty	Major Questions	Potential Uncertainties
Business	Will political, economic, labor, social, technological, environmental, legal or, other factors adversely affect the business environment?	Changes in political viewpoint (e.g., elections) Economic disruptions (e.g., recession). Global disruptions (e.g., supply chain). Changes to laws and regulations. Disruptive technologies. Adverse publicity.
Market	Will there be a market if the product or service works?	User and consumer demand. Threats from competitors (quality and price) and adversaries (e.g., hackers and terrorists). Continuing stakeholder support.
Management	Does the organization have the people, processes, and culture to manage a major system?	Organization culture. SE and management experience and expertise. Mature baselining processes (technical, cost, schedule). Reliable cost estimating processes.
Performance (Technical)	Will the product or service meet the required desired performance?	Defining future requirements in dynamic environments. Understanding of the technical baseline. Technology maturity to meet performance. Adequate modeling, simulation, test, and evaluation capabilities to predict and evaluate performance. Availability of enabling systems needed to support use.
Schedule	Can the system that provides the product or service be delivered on time?	Concurrency in development. Impact of uncertain events on schedule. Time and budget to resolve technical and cost risks.
Development and Production Cost	Can the system be delivered within the budget? Will the cost be affordable?	Changes in missions. Technology maturity. Hardware and software development processes. Industrial/supply chain capabilities. Production facilities capabilities and processes.
Operations and Support Cost	Can the owner afford to operate and support the system? Will the cost be affordable?	Increasing operations and support (e.g., resource or environmental) costs. Resiliency of the design to new missions and tasks. Changes in maintenance or logistics strategy/needs.
Sustainability	Will the system provide sustainable future value?	Availability of future resources and impact on the natural environment.

Sources of Uncertainty and Risk. There are many sources of epistemic uncertainty that impact SE in the system life cycle. Table 1.4 provides a partial list of some of the major uncertainties that confront project managers and SE practitioners and describes some of the implications for SE.

Decisions Under Uncertainty

As can be seen from Table 1.4, uncertainties impact every SE decision process. Taking decisions before having enough knowledge is potentially very risky. Key decisions that have a strong impact on the solution require reducing uncertainty by closing the knowledge gap to an appropriate level. However, SE practitioners must be able to make decisions under uncertainty and should record a corresponding risk with those decisions (see Sections 2.3.4.3 and 2.3.4.4).

1.4.2 Cognitive Bias

SE practitioners need to obtain information from stakeholders throughout the system life cycle. SE practitioners and stakeholders (individual or groups) are subject to cognitive biases when interpreting uncertain information. The best defense from cognitive biases is understanding what they are and how they can be avoided and setting up organizational projects to obtain unbiased assessments. Cognitive biases are mental errors in judgment under uncertainty caused by our simplified information processing strategies (sometimes called heuristics) and are consistent and predictable (Tversky and Kahneman, 1974). There are many lists of cognitive biases, including one that lists 50 sources (Hallman, 2022). Cognitive biases can affect both individual and teams of SE practitioners (McDermott, et al., 2020). Cognitive biases can contribute to incidents, failures, or disasters as a result of distorted decision making and can lead to undesirable outcomes. Cognitive biases are included in a field called Behavioral Decision-Making. Table 1.5 lists some of the most common cognitive biases.

For major systems decisions, more formal methods are required to avoid cognitive biases. Both Tversky and Kahneman (1974) and Thaler and Sunstein (2008) describe mitigation methods suitable to different environments. The most effective methods are external group methods. For example, NASA (2003) recommends the Independent Technical Authority (ITA) to warn decision makers of the potential for failure. The ITA must be both financially and organizationally independent of the project manager. Another method, adopted by the aviation industry, is called the Crew Resource Management (CRM) method. With the CRM method, all crew members, including the co-pilot, are responsible for warning the pilot of imminent danger.

1.4.3 Systems Engineering Principles

SE is a relatively young discipline. The emergence of a set of SE principles has occurred over the past 30 years within the discipline. In reviewing various published SE principles, a set of criteria emerged for SE principles. SE principles cover broad application within the practice; they are not constrained to a particular system type, to the system development or operational context, or to a particular life cycle stage. SE principles transcend these system characteristics and inform a worldview of the discipline. Thus, a SE principle:

- transcends a particular life cycle model or stage,
- transcends system types,
- transcends a system context,
- informs a world view on SE,
- is not a “how to” statement,
- is supported by literature or widely accepted by the community (i.e., has proven successful in practice across multiple organizations and multiple system types),
- is focused, concise, and clearly worded.

SE principles are a form of guidance proposition which provide guidance in application of the SE processes and a basis for the advancement of SE. SE has many kinds of guidance propositions that can be classified by their sources, e.g., heuristics (derived from practical experience as discussed in Section 1.4.4), conventions (derived from social agreements), values (derived from cultural perspectives), and models (based on theoretical mechanisms). Although these all support purposeful judgment or action in a context, they can vary greatly in scope, authority, and conferred capability. They can all be refined, and as they mature, they gain in their scope, authority, and capability, while the set becomes more compact. A key moment in their evolution occurs with gaining insight into why they work, at which point they become principles. Principles can have their origins associated in referring to them as “heuristic principles,” “social principles,” “cultural principles,” and “scientific principles,” although in practice it is usually sufficient to just refer to them as SE principles. SE principles are derived from principles of these various origins providing a diverse set of transcendent principles based on both practice and theory.

TABLE 1.5 Common cognitive biases

Cognitive Bias	Description	Implication for the SE Practitioner.
Framing	How we ask the question or describe the decision matters.	Carefully word questions and problem description to avoid influencing the response.
Representativeness	People draw conclusions based on representative characteristics and often ignore relevant facts or the base rates.	Discuss the relevant facts and data before requesting a judgment about an uncertainty or risk. Use Bayes Law to update our beliefs after we receive new data. Teams that reflect Diversity, Equity, and Inclusion principles can help reduce the bias for the team (see Section 5.2).
Availability	We place too much weight on vivid, striking, and recent events.	Ask about the relevant facts and data before requesting a judgment about an uncertainty or risk. Design systems to provide the relevant data.
Anchoring	The initial estimate affects the final estimates.	Never begin by asking about the expected outcome. Instead obtain information about the worst or best outcomes first to understand the range of outcomes.
Motivational	When making probability judgments, people have incentives to provide estimates that will benefit themselves	Understand the potential bias of an individual providing an assessment. For example, a technology developer has an incentive to overestimate technology readiness if a more conservative estimate could result in loss of funding.
Optimism	We overestimate the likelihood of good outcomes and underestimate the likelihood bad outcomes.	Seek data on similar bad outcomes. Obtain assessments from experts not involved in the decision.
Confirmation	We seek or put more weight on data that confirms our beliefs.	Actively seek data that would disprove our current belief in all tests and evaluations.
Group Think	A group of people make irrational or unsound decisions to suppress dissent and maintain group harmony.	Seek dissenting opinions inside the group and seek outside assessments.
Authority	We trust and are more often influenced by the opinions of people in positions of authority	Assess the opinion independent of the source.
Rankism	Assumption that person of higher rank is always correct in decisions	Seek to determine correct decision

In addition, SE principles differ from systems principles in important ways (Watson, et al., 2019). System principles address the behavior and properties of all kinds of systems, looking at the scientific basis for a system and characterizing this basis in a system context via specialized instances of a general set of system principles. SE principles build on systems principles that are general for all kinds of systems (Rousseau, 2018) (Watson, 2020) and for all kinds of human activity systems (Senge, 1990) (Calvo-Amodio and Rousseau, 2019).

INCOSE compiled an early list of principles consisting of 8 principles and 61 subprinciples in 1993 (Defoe, 1993). These early principles were important considerations recognized in practice for the success of system developments and ultimately became the basis for the SE processes. These early principles were focused on particular aspects of the

SE process and particular life cycle stages. The INCOSE work on SE principles considered these earlier sources and compiled a set of SE principles that are transcendent. The INCOSE SE Principles (2022) documents each SE principle with a description, evidence that supports the principle (e.g., observable evidence of the application, proof from scientific evidence), and implications in SE practice for application of the principle. There are presently 15 SE principles and 20 subprinciples as shown in Table 1.6.

TABLE 1.6 SE principles and subprinciples

1	SE in application is specific to stakeholder needs, solution space, resulting system solution(s), and context throughout the system life cycle.
2	SE has a holistic system view that includes the system elements and the interactions amongst themselves, the enabling systems, and the system environment.
3	SE influences and is influenced by internal and external resources, and political, economic, social, technological, environmental, and legal factors.
4	Both policy and law must be properly understood to not over-constrain or under-constrain the system implementation.
5	The real system is the perfect representation of the system.
6	A focus of SE is a progressively deeper understanding of the interactions, sensitivities, and behaviors of the system, stakeholder needs, and its operational environment.
	Sub-Principle 6(a): Mission context is defined based on the understanding of the stakeholder needs and constraints
	Sub-Principle 6(b): Requirements and models reflect the understanding of the system
	Sub-Principle 6(c): Requirements are specific, agreed to preferences within the developing organization
	Sub-Principle 6(d): Requirements and system design are progressively elaborated as the development progresses
	Sub-Principle 6(e): Modeling of systems must account for system interactions and couplings
	Sub-Principle 6(f): SE achieves an understanding of all the system functions and interactions in the operational environment
	Sub-Principle 6(g): SE achieves an understanding of the system’s value to the system stakeholders
	Sub-Principle 6(h): Understanding of the system degrades during operations if system understanding is not maintained.
7	Stakeholder needs can change and must be accounted for over the system life cycle.
8	SE addresses stakeholder needs, taking into consideration budget, schedule, and technical needs, along with other expectations and constraints.
	Sub-Principle 8(a): SE seeks a best balance of functions and interactions within the system budget, schedule, technical, and other expectations and constraints.
9	SE decisions are made under uncertainty accounting for risk.
10	Decision quality depends on knowledge of the system, enabling system(s), and interoperating system(s) present in the decision making process.
11	SE spans the entire system life cycle.
	Sub-Principle 11(a): SE obtains an understanding of the system
	Sub-Principle 11(b): SE defines the mission context (system application)
	Sub-Principle 11(c): SE models the system
	Sub-Principle 11(d): SE designs and analyzes the system
	Sub-Principle 11(e): SE tests the system
	Sub-Principle 11(f): SE supports the production of the system
	Sub-Principle 11(g): SE supports operations, maintenance, and retirement
12	Complex systems are engineered by complex organizations.
13	SE integrates engineering and scientific disciplines in an effective manner.
14	SE is responsible for managing the discipline interactions within the organization.
15	SE is based on a middle range set of theories.
	Sub-Principle 15 (a): SE has a systems theory basis
	Sub-Principle 15 (b): SE has a physical logical basis specific to the system
	Sub-Principle 15 (c): SE has a mathematical basis
	Sub-Principle 15 (d): SE has a sociological basis specific to the organization

From INCOSE SE Principles (2022). Usage per the INCOSE Notices page. All other rights reserved.

These principles provide a start in defining a transcendent disciplinary basis for SE. Application of the principles aids in determining a system life cycle model, implementing SE processes, and defining organizational constructs to help the SE practitioner successfully develop and sustain the SoI.

1.4.4 Systems Engineering Heuristics

Summary Heuristics provide a way for an established profession to pass on its accumulated wisdom. This allows practitioners to gain insights from what has been found to work well in the past, and apply the lessons learned. Heuristics usually take the form of short expressions in natural language. These can be memorable phrases encapsulating shortcuts, "rules of thumb," or "words of the wise," giving general guidelines on professional conduct or rules, advice, or guidelines on how to act under specific circumstances. Heuristics usually do not express all there is to know, yet they can act as a useful entry point for learning more. At their best, heuristics can act as aids to decision making, value judgments, and assessments.

Interest in SE heuristics currently centers on their use in two contexts: (1) encapsulating engineering knowledge in an accessible form, where the underlying practice is widely accepted and the underlying science understood, and (2) overcoming the limitations of more analytical approaches, where the science is still of limited use. This is especially applicable as we extend the practice of SE to providing solutions to inherently complex, unbounded, ill-structured, or very difficult problems.

Background Engineering first emerged as a series of skills acquired while transforming the ancient world, principally through buildings, cities, infrastructure, and machines of war. Since then, mankind has sought to capture the knowledge of "how to" to allow each generation to learn from its predecessors, enabling more complex structures to be built with increasing confidence while avoiding repeated real-world failures. For example, early cathedral builders encapsulated their knowledge in a small number of "rules of thumb," such as "maintain a low center of gravity" and "put 80% of the mass in the pillars." Designs were conservative, with large margins. When the design margins were exceeded (e.g., out of a desire to build higher and more impressive structures), a high price was sometimes paid, with the collapse of a roof, a tower, or even a whole building. From such failures, new empirical rules emerged. Much of this took place before the science behind the strength of materials or building secure foundations was understood. Only in recent times have computer simulations revealed the contribution toward certain failures played by such dynamic effects as wind shear on tall structures.

Since then, engineering and applied sciences have co-evolved: with science providing the ability to predict and explain performance of engineered artefacts with greater assurance and engineering developing new and more complex systems, requiring new scientific explanations and driving research agendas. In the modern era, complex and adaptive systems are being built which challenge conventional engineering sciences, and we are turning to social and behavioral sciences, management sciences, and increasingly systems science to deal with some of the new forms of complexity involved and guide the profession accordingly.

Current Use Renewed interest in the application of heuristics to the field of SE stems from the seminal work of Maier and Rechtin (2009), and their book remains the best single published source of such knowledge. Their motivation was to provide guidance for the emerging role of system architect as the person (or team) responsible for coordinating engineering effort toward devising solutions to complex problems and overseeing their implementations. They observed that it was in many cases better to apply heuristics than attempt detailed analysis. The reason for this is the number of variables involved and the complexity of the interactions between stakeholders, internal dynamics of system solutions, and the organizations responsible for their realization. Some examples of SE heuristics are:

- ***Don't assume that the original statement of the problem is necessarily the best, or even the right one.*** This has to be handled with tact and respect for the user, but experience shows that failure to reach mutual understanding early on is a fundamental cause of failure, and strong relationships forged in the course of doing such coordination with stakeholders can pay off when solving more difficult issues which might arise later on.

- ***In the early stages of a project, unknowns are a bigger issue than known problems.*** Sometimes developing a clear understanding of the environment, all of the stakeholders, and the ramifications of possible solutions uncovers many unanticipated issues.
- ***Model before build, wherever possible.*** System Science postulates "The only complete model of the system in its environment is the system in its environment," which leads into using evolutionary life cycles, rapid deployment of prototypes, agile life cycles, and so on. This heuristic opens a door into twenty-first-century systems.

A repository of heuristics can act as a knowledge base, especially if media (such as video clips or training materials) or even interactive media (to encourage discussion and feedback) are included. A heuristics repository should link to other established knowledge sources and be tagged with other metadata to allow flexible retrieval. It should be organized to reflect accepted areas of SE competency and allow users to assemble a personal set of heuristics most meaningful to them, being relevant to their professional or personal sphere of activity.

1.5 SYSTEM SCIENCE AND SYSTEMS THINKING

This section considers the nature and relationship between systems science and systems thinking and describes how they relate to SE.

Relationship between Systems Science, Systems Thinking, and SE

The association of concepts such as system, boundary, relationships, environment/context, hierarchy, emergence, communication, and control, among others, when interrelated with purpose, gives rise to a *systems worldview* (Rousseau, et al., 2018). Interrelating concepts with purpose changes how we investigate and reason about things, producing *systems thinking*. Systems thinking enables us to recognize systems patterns across different phenomena, problem contexts, and disciplines. Studying these patterns has produced the systems sciences of General System Theory, Cybernetics, and Complexity Theory and their related systems methodologies, models, and methods. The application of systems thinking and systems science concepts, principles, methodologies, models, and methods in engineering is one of the bases for the practice of SE. Applying SE, and reflecting on the results, help us improve systems science and systems thinking, further enhancing our ability to design and intervene in complex systems—a virtuous cycle. Through this virtuous cycle, we develop principles to better our SE applications (Rousseau, et al., 2022).

Figure 1.8 depicts this virtuous cycle as a multifaceted and purposeful activity to deliver elegant solutions to complex problems, supported by principles that guide why, what, and how we do SE. To connect our purpose to our actions, we adopt a systemic approach, because complexity and elegance are both systems phenomena. Our systemic approach is of course guided by our systems principles. The kinds and relationships of principles, as well as how they inform and are informed by SE practice, is depicted in Figure 1.8. We select and organize these based on our intentions as expressed by our motivational principles. We use our transdisciplinary principles to select and organize our technique principles. In this way, the systemic relationships between our principles support how our principles guide the systemic relationships between our purpose, approach, and practice. The systemic roles our principles play in our discipline thus support the systematic evolution of our value in society.

The success of SE applications reinforces the credibility of the systems worldview which in turn enhances the SE practitioner's ability to conceptualize why a solution is needed, how a solution can be conceptualized, and what tools and/or methods to use to solve complex problems and achieve elegant solutions.

Systems Science

Questions about the nature of systems, organization, and complexity are not specific to the modern age. As Warfield (2006) put it:

> Virtually every important concept that backs up the key ideas emergent in systems literature is found in ancient literature and in the centuries that follow.

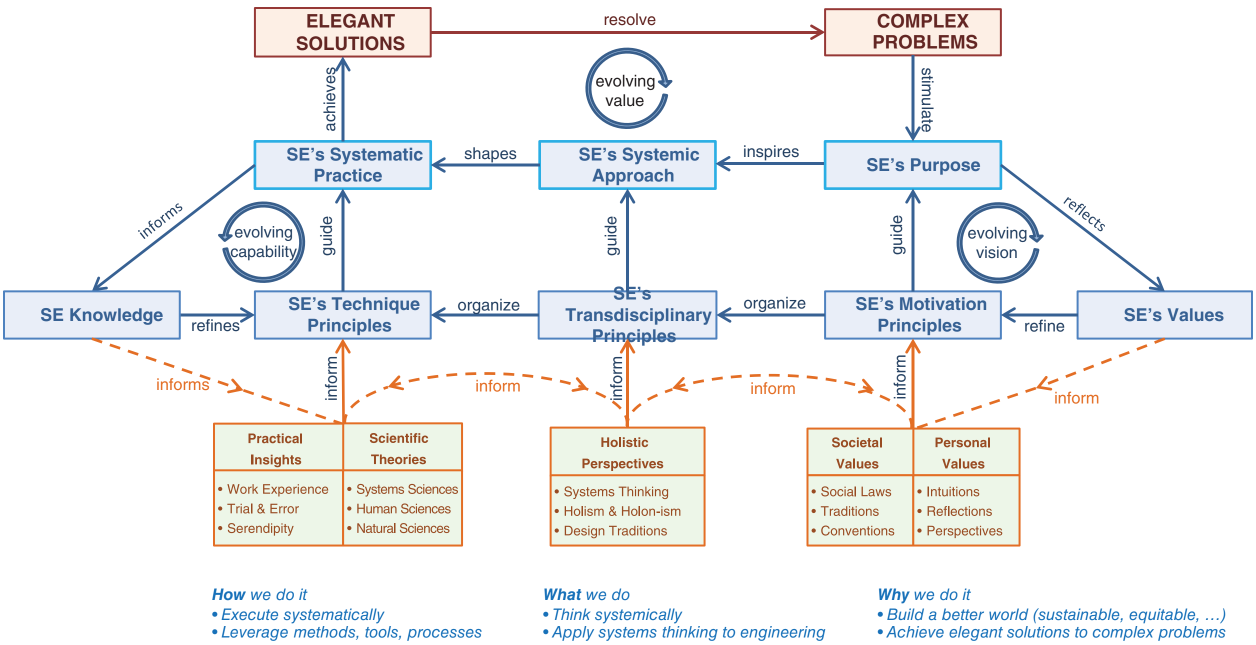

FIGURE 1.8 An architectural framework for the evolving the SE discipline. From Rousseau, et al. (2022). Used with permission. All other rights reserved.

Systems science can be defined as a transdisciplinary approach interested in understanding all aspects of systems with the goals of (1) identifying, exploring, and understanding patterns of behavior crossing disciplinary fields and areas of application, and (2) establishing a general theory applicable to all types of systems whether physical, natural, engineered, or social. Attempts to establish a systems science have taken on both reductionist and holistic forms, and both are valuable. For instance, a clock is a system, but its workings can be explained through reductionism. On the other hand, a holistic approach helps us understand why we need clocks, how clocks exist (operate/sustain/degrade) in their environment throughout their life cycle, and how the esthetics of their design evolve over time. Both the reductionist and the holistic approaches to explanation involve systemic arguments but each starts from different directions—bottom-up and outside-in. Complexity Theory has had some success developing a science of systems using a reductionist approach. Agent-based modeling, pioneered at the Santa Fe Institute, works from the "bottom-up" and seeks to explain the behavior of whole systems in terms of the rules of interaction of the "agents" that constitute the system.

Where reductionist (traditional) methods prove unsuccessful, systems science relies on the holistic approach. A holistic approach is adept at connecting and contextualizing systems, system elements, and their environments to understand difficult to explain patterns of organized complexity. This was the approach taken by Ludwig von Bertalanffy in developing General System Theory and Norbert Wiener in developing Cybernetics. The biologist von Bertalanffy was one of the first to argue for a general science of systems. He explained the scientific need for systems-based research as an alternative to traditional analytical procedures in science. This alternative method would overcome the limitations that result from explaining a system by breaking it down to its constituent parts and then being reconstituted from its parts, either materially or conceptually:

> This is the basic principle of *classical* science, which can be circumscribed in different ways: resolution into isolable causal trains or seeking for *atomic* units in the various fields of science, etc. (von Bertalanffy, 1969)

This makes it impossible to account for the emergent properties that systems display as a result of the interrelationships between their parts (see Section 1.3.2). Instead, von Bertalanffy promoted an alternative worldview concerned with the laws that apply to systems behavior in general. Such a General System Theory was possible, von Bertalanffy thought, and would be particularly valuable, because of the large number of parallelisms that appear across systems independent of the types and quantities of system elements in the systems:

> Thus, there exist models, principles, and laws that apply to generalized systems or their subclasses, irrespective of their particular kind, the nature of their component elements, and the relations or 'forces' between them. It seems legitimate to ask for a theory, not of systems of a more or less special kind, but of universal principles applying to systems in general. In this way we postulate a new discipline called General System Theory. (von Bertalanffy, 1971)

The study of general systems was to focus on such principles as:

> growth, regulation, hierarchical order, equifinality, progressive differentiation, progressive mechanization, progressive centralization, closed and open systems, competition, evolution toward higher organization, teleology, and goal-directedness. (Hammond, 2003)

The systems sciences, including General Systems Theory, Cybernetics, and Complexity Theory, seek to provide key foundational concepts to build a common language and intellectual foundations to make rigorous systems theories and tools accessible to practitioners. Where they succeed, they can serve as the foundation for a meta-discipline such as SE, transdisciplinary in nature, that unifies scientific and engineering practices. SE, informed by systems science, would be in a powerful position to enhance its theory and practice in ways that would make it applicable to the most complex of systems.

Finally, identifying SE's principles and heuristics can offer a useful approach to categorize systems-related knowledge and to focus research efforts. Systems principles and heuristics are special cases of guiding propositions. A guiding proposition provides guidance for purposeful judgment or action in a context and offers a wider perspective to that of a principle or a heuristic. Guiding propositions vary in (1) scope—the range of SE contexts they work, (2) authority—how compelling they are, and (3) capability—how predictable the outcomes of applying them are (Rousseau, et al., 2022). Readers can consult details on SE Principles in Section 1.4.3 and details on SE Heuristics in Section 1.4.4.

Systems Thinking Divergences

Systems thinking is a key enabler of SE. It is one of the core competencies defined in INCOSE SE Competency Framework (2018). Systems thinking applies the properties, concepts, and principles of systems to the given situation as a framework for curiosity—to get insight and understanding about the situation.

There needs to be a balance between the being systematic with the application of SE processes (as described in Part II) and being systemic, applying systems thinking to drive these processes. As SE practitioners, it is vital to possess the knowledge and skills necessary to perform holistic analysis and guide systemic intervention. Systems thinking lacks a unified definition; however, the following captures the nature of systems thinking and some key ideas:

> Systems thinking is a field characterized by a baffling array of methods and approaches. We posit that underlying all, however, are four universal rules called DSRP (distinctions, systems, relationships, and perspectives). We make distinctions between and among things and ideas, each implying the existence of another. We identify systems, which are composed of parts and wholes. We recognize relationships composed of actions and reactions. We take perspectives consisting of a point (from which we see) and a view (that which is seen). (Cabrera, et al., 2015)

This definition incorporates aspects of complex problem situations, such as “distinctions” and “perspectives,” which it is essential to take account of, but which systems science may never be able to incorporate into its scientific models.

Based on the pioneering work of Ludwig von Bertalanffy in General System Theory, Norbert Wiener in Cybernetics, Jay Forrester in System Dynamics, Peter Checkland in Soft Systems Thinking, and others, a variety of systems methodologies, models, and methods have been formalized to perform systemic analyses and interventions. The SE practitioner can take advantage of this diversity providing they are aware of what the different methodologies, models, and methods do well, and what they are less good at. To assist systems thinking practitioners in selecting the most appropriate systems approaches, Jackson and Keys (1984) offered an initial classification of systems methodologies, the Systems of Systems Methodologies (SOSM), according to their strengths in addressing the complexity of systems and in reconciling divergences among stakeholder viewpoints. Jackson (2019) has since updated the SOSM, to reflect developments in Complexity Theory, by incorporating lessons from the Cynefin framework (Kurtz and Snowden, 2003). This use of different systems approaches in informed combinations, according to their strengths and weaknesses and the nature of the problem situation, is called Critical Systems Thinking (CST) (Jackson, 2003, 2019). CST is a multi-perspectival, multi-methodological, and multi-method approach.

While most of the prominent systems thinking approaches are rooted and/or contextualized within the management sciences, these approaches apply equally to SE practice. This is because the problems faced by SE practitioners, such as the need to incorporate cultural, social, political, and project management perspectives into systems models and other SE tasks, are common to the management sciences.

According to Jackson (2019), systems methodologies translate hypotheses about the nature of problem situations, and how they can be improved, into practical action. There are a number of systems methodologies available, for example, system dynamics, the viable system model, soft systems methodology, and critical systems heuristics. Each is based upon different assumptions about the world and how best to intervene in it. Together, these methodologies can recognize and respond to the range of issues encountered during the exploration of complex problem situations. These systems approaches can then be used, individually or in combination, in the problem situation. When the systems approaches are used in combination, the weighting of each system approach in the hybrid solution will be tailored based on the technical, organizational, cultural, and political factors within the organization and the relative dominance of those factors. According to systems thinkers, if SE can embrace the full range of systems methodologies, models, and methods, it will be in a much better position to tackle the hyper-complexity plaguing projects, organizations, and society in the contemporary world.