Measuring Complexity of Engineering Systems:
A System State and Behavioral Approach

by Farouk Harold Bonilla-Mendoza

B.S. in Electrical Engineering, 1990, National University of Colombia - Bogotá
M.S. in Electrical Engineering, 1998, University of Puerto Rico - Mayagüez
M.S. in Imaging Science, 2008, Rochester Institute of Technology

A Dissertation submitted to

The Faculty of
The School of Engineering and Applied Science
of The George Washington University
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

August 31, 2021

Dissertation directed by

Thomas Holzer
Professional Lecturer of Engineering Management and Systems Engineering

Shahryar Sarkani
Professional Lecturer of Engineering Management and Systems Engineering

The School of Engineering and Applied Science of The George Washington University certifies that Farouk Harold Bonilla-Mendoza has passed the Final Examination for the degree of Doctor of Philosophy as of July 29, 2021. This is the final and approved form of the dissertation.

Measuring Complexity of Engineering Systems:
A System State and Behavioral Approach

Farouk Harold Bonilla-Mendoza

Dissertation Research Committee:

Thomas Holzer, Professional Lecturer of Engineering Management and Systems Engineering; Dissertation Co-Director

Shahryar Sarkani, Professional Lecturer of Engineering Management and Systems Engineering; Dissertation Co-Director

Thomas Mazzuchi, Professor of Engineering Management and Systems Engineering and Decision Sciences, Committee Member

Shahram Sarkani, Professor of Engineering Management and Systems Engineering, Committee Member

Amir Etemadi, Associate Professor of Engineering and Applied Science, Committee Member

© Copyright 2021 by Farouk Harold Bonilla-Mendoza
All rights reserved

Dedication

The author wishes to dedicate this dissertation to his family: his wife Enid and his children Leslie, Nathan, and Mario, for their love, patience, sacrifice, and support. He would also like to thank his parents Mario and Ana, for inspiring him and believing in him throughout his whole life, and his siblings Ruby, Armando, Mario, and Marhsy, for their understanding and enthusiasm.

Acknowledgments

The author wishes to acknowledge his daughter Leslie, who encouraged and assisted him when most needed, making it possible to bring this project to fruition. Much gratitude also to the author's academic advisors Drs. Holzer and Sarkani for their invaluable reviews, inputs, ideas, insightful discussions, and guidance.

Abstract of Dissertation

Measuring Complexity of Engineering Systems: A System State and Behavioral Approach

System complexity has grown exponentially over the last three decades, but the development of complexity measures to quantify system complexity is still in its infancy. This research proposes a comprehensive measure and methodology to quantify the complexity of engineering systems. The complexity measure and associated methodology are based on a holistic system approach and formulate an engineering system's complexity as a single quantifiable system property that simultaneously incorporates system structure, function, states, and behavior. Most current measures only quantify either structural complexity or functional complexity. However, the proposed comprehensive complexity measure includes both structure and function, and also states and behavior, enabling the objective assessment of different engineering designs, identification of elements driving significant complexity, and management of system complexity.

Complex engineering systems are characterized by large structures with complicated configurations consisting of numerous components and interconnections. The interrelations among components become very important, and new functionality and system states may develop, challenging accurate predictions of system behavior. Structure, function, states, and behavior are critical characteristics to consider in assessing modern systems' complexity. These system characteristics are intertwined and have a profound impact on system design and future system performance. The proposed

complexity measure is derived from a range of disciplines and principles, including quantum mechanics, signal processing, and systems engineering. This dissertation contains two fundamental parts. In the first, the proposed complexity measure is formulated, following a mathematical modeling methodology. In the second, a system model is developed, and the proposed complexity measure is applied to assess the system model's complexity. An ATM system model was created using a Model-Based Systems Engineering (MBSE) modeling tool. A simulation study was performed to obtain the output data consisting of the values of the complexity quantity associated with the system. Systems engineering professionals can apply the proposed measure to quantify system complexity and, therefore, identify complex elements and compare different system designs or different systems. The proposed complexity measure, integrated with the modeling methodology, enables these professionals to get a more comprehensive understanding of system complexity. The complexity measure provides a solid basis for evaluating and communicating complexity during system design and architectural modeling and improving scheduling and cost estimates during system development.

Table of Contents

Dedication ..... **iv**

Acknowledgments ..... **v**

Abstract of Dissertation ..... **vi**

List of Figures ..... **xi**

List of Tables ..... **xii**

List of Acronyms ..... **xiii**

Glossary ..... **xiv**

Chapter 1: Introduction **1**

1.1 Background 1

1.2 Research Motivation 2

1.3 Problem Statement 4

1.4 Research Questions 4

1.5 Research Objectives 5

1.6 Research Hypotheses 5

1.7 Research Contributions 6

1.8 Dissertation Outline 7

Chapter 2: Literature Review **9**

2.1 Systems and Complex Systems 9

2.1.1 System Structural Representation and Network Theory 11

2.1.2 Design Structured Matrix and Adjacency Matrix 13

2.1.3 Requirements, system adaptation, and system behavior 14

2.2 Life-Cycle 15

2.3 System Architecture Process 18

2.4 Models and Model-Based System Engineering 19

2.5 The Object-Oriented Systems Engineering Method 23

2.6 Information Theory 24

2.7 Molecular Quantum Mechanics Theory Overview 26

2.8 System Complexity Definition 29

2.9 Static and Dynamic Complexity Measures 30

2.10 Engineering Design and Complexity Measures 32
2.10.1 The Design Problem 33
2.10.2 The Design Process 33
2.10.3 The Design Product 34
2.10.4 Remarks on Engineering Design Complexity Measures 36

2.11 System Structural and Functional Complexity Measures 38
2.11.1 A Disjointed Structural and Functional Complexity Measure 38
2.11.2 A Molecular Quantum Mechanics-Based Structural Complexity Measure 39

2.12 A Spectral Complexity Measure 45
2.12.1 The Laplacian and normalized Laplacian spectral complexity measures ... 46
2.12.2 Natural Connectivity and Laplacian Natural Connectivity Spectral Complexity Measures 46

2.13 Summary 48

Chapter 3: Research Methodology 49

3.1 Development of Complexity Metric 50
3.1.1 Theoretical Framework 51
3.1.2 System Complexity Measure Incorporating System Behavior 52
3.1.3 Developing a Model for a Test System using an MBSE Approach 59
3.1.4 Advantages of The Proposed Complexity Measure over Traditional Complexity Measures. 67

Chapter 4: Results 71

4.1 Test Case I: Baseline 71

4.2 Test Case II: Change of Requirements 76

4.3 Summary 84

Chapter 5: Conclusions, Limitations, and Recommendations 87

5.1 Conclusions 87

5.2 Limitations 89

5.3 Recommendations for Future Research 91

References 95

List of Figures

Figure 1. Classical representation of an open system. 10
Figure 2. Systems with different structural complexity. 12
Figure 3. The benzene molecule, an example of a conjugated π-system. 40
Figure 4. Example of a network walk 47
Figure 5. ATM SD of Withdrawal from the Checking Account (©2021 IEEE). 60
Figure 6. SD – Deposit to the Checking Account (©2021 IEEE). 61
Figure 7. SD -Transfer from Checking Account to Saving Account (©2021 IEEE). 62
Figure 8. ATM System BDD (©2021 IEEE). 63
Figure 9. ATM State Machine Diagram (©2021 IEEE). 65
Figure 10. ATM BDD, including new requirements (©2021 IEEE). 77
Figure 11. Modified ATM SMD with new requirements (©2021 IEEE) 78
Figure 12. ATM SD's Withdrawal from the Checking Account with new requirements (©2021 IEEE). 79
Figure 13. ATM SD's Deposit from the Checking Account with new requirements 80
Figure 14. ATM SD's Transfer from the Checking Account with new requirements. 81

List of Tables

Table 1. Number of Functions per Component (Withdrawal) 74
Table 2. Number of Functions per Component (Deposit) 74
Table 3. Number of Functions per Component (Transfer) 74
Table 4. Number of Functions per Component (Withdrawal - New requirements) 82
Table 5. Number of Functions per Component (Deposit– New requirements) 82
Table 6. Number of Functions per Component (Transfer – New requirements) 82

List of Acronyms

BDD	Block Definition Diagram
DSM	Design Structure Matrix
HMO	Hückel's Molecular Orbital theory or method
HW	Hardware
IBD	Internal Block Diagram
MQM	Molecular Quantum Mechanics
OMG	Object Management Group
OO	Object-Oriented software
OOSEM	Object-Oriented Systems Engineering Method
SD	Sequence Diagram
SME	Subject Matter Expert
SOI	System Of Interest.
SoS	System of Systems
STM	State Machine Diagram
SW	Software
SysML	Systems Modeling Language
UML	Universal Modeling Language
V&V	Verification and Validation

Glossary

Activity	"A set of cohesive tasks of a process." [26]
Agent	"A human user or users who interacts with the system or parts thereof." [14]
Architecture	"Fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution." [1]
Arrival Process	"The arrival process for infinite-population models is usually characterized in terms of interarrival times of successive customers. Arrivals may occur at scheduled times or at random times. When at random times, the interarrival times are usually characterized by a probability distribution. In addition, customers may arrive one at a time or in batches. The batch may be of constant size or of random size." [77]
Attribute	"An inherent property or characteristic of an entity that can be distinguished quantitatively or qualitatively by human or automated means." [1]
Bayes's Rule	"Bayes's rule provides a means for the computation of posterior probabilities ($P(A_i|B)$, also known as a posteriori -after the fact- probabilities) from the known prior probabilities ($P(A_i)$, also known as a priori probabilities) and the conditional probabilities $P(B|A_i)$ for a particular situation or experiment." [43]
Behavior	"A sequence of functions (and associated changes in state) that form executes in order to deliver value, and it is a product attribute." [15]
Behavioral feature	"Along with structural features, blocks can also own behavioral features that describe which requests a block can respond to. A behavioral feature may have an associated method that is a behavior invoked when the block handles a request for the feature." [27]
Beta Distribution	"A random variable *Y* is said to have a standard beta distribution if its probability density function is given by:

$$f_Y(Y) = \begin{cases} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}(y)^{\alpha-1}(1-y)^{\beta-1}, 0 < y < 1 \\ 0, otherwise \end{cases}$$

where α and β determine the shape of the density function and $\Gamma(\alpha)$ is the gamma function of the argument α." [43].

Black box	"Black box represents an external view of the system (attributes)." [26]
Block	"A very general modeling concept in SysML that is used to model entities that have structure such as systems, hardware and equipment, software, or physical object." [27]
Boundary	"(Also known as membrane) separates the system from the external world. It serves to concentrate interactions inside the system while allowing exchange with external systems." [1]
Complex	"An adjective describing a system's design or code that is difficult to understand because of numerous components or relationships among components." [1]
Complexity	"A measure of how difficult it is to understand how a system will behave or to predict the consequences of changing it." [20]
Conjugated π-systems	"A special class of polyatomic molecules containing π-bonded atoms, especially the conjugated polyenes and aromatic molecules based on benzene." "When the system is conjugated (i.e., there are orbitals on several neighboring atoms capable of forming an extended system of π-orbitals, as in benzene), the simplest description of the bonding is in terms of the HMO method. This takes full account of the symmetry of the networks of atoms through which the π-bonds extend (e.g., the planar hexagonal symmetry of the benzene), and leads to expressions for the relative energies of the different molecular orbital in terms of a single parameter." [31]
Design Structure Matrix	"Network modeling tool used to represent the element comprised in a system and their interactions, thereby highlighting the system's architecture (or design structure)." [46]
Emergent behavior	"A behavior of the system that cannot be understood

	exclusively in terms of the individual system elements." [26]
Engineering System	"A man-made aggregation which may contain physical, informational, human, natural, and social elements; it is normally created by people." [6]
Entropy	"A measure of uncertainty before an event occurs." [49]
Environment	"The surroundings (natural or man-made) in which the SOI is utilized and supported or in which the system is being developed, produced and retired." [26]
Event	"Set of circumstances that cause an instantaneous change in the state of the system." [77]
Failure	"The event in which any part of an item does not perform as required by its specification, that is, past design limits or beyond the margin of safety." [26]
Graph	"A graph G is an ordered triple (V(G), E(G), ψG) consisting of a nonempty set V(G) of vertices, a set E(G), disjoint from V(G), of edges, and an incidence function ψG that associates with each edge of G an unordered pair of (not necessarily distinct) vertices of G. If e is an edge and u and v are vertices such that ψG (e)=uv, then e is said to join u and v; the vertices u and v are called the ends of e." [33]
Graph Theory	A branch of mathematics (also known as "topology") that studies network properties. [33]
Holism	"A system should be considered as a single entity, a whole, not just as a set of parts." [1]
Hermitian	Matrix of a transformation that is equal to its conjugate transpose. [40]
Interaction	"The properties, capabilities, and behavior of a system are derived from its parts, interactions between those parts, and interactions with other systems." [1]
Interface	"A shared boundary between two functional units, defined by functional characteristics, signal characteristics, or other characteristics, as appropriate." [26]
Life-cycle model	"A framework of processes and activities concerned with the life-cycle, which also acts as a common reference for

	communication and understanding." [26]
Methodology	"Collection of related processes, methods, and tools used to support a specific discipline." [26]
Method	"Some behaviors owned by the block only execute in response to a particular stimulus, specifically when a request is made via a provided behavioral feature (operation or reception). Such a behavior is called a method, and it is related to the behavioral feature that was requested." [27]
Model	"Simplified representation of a system. It can be conceptual, verbal, diagrammatic, physical, or formal (mathematical)." [62].
Network	"Fundamental topology for systems that forms the basis of togetherness, connection, and dynamic interaction of parts that yield the behavior of complex systems." [1]
Open Systems	"Systems that interact significantly with its environment, allowing energy, information, or matter to cross its boundary." [22]
Operation	"Behavioral feature that is typically triggered by a synchronous request (i.e., when a requester waits for a response). Each operation defines a set of parameters that describes the arguments passed back out once a request has been handled, or both. Note that an operation may be triggered by an asynchronous request (i.e., when the requester does not wait). In that case, no arguments are passed back to the requester." [27]
Object Management Group	The Object Management Group (OMG) is an international non-profit technology standards consortium founded in 1989. The Object Management Group standards include the Model Driven Architecture (MDA), the Universal Modeling Language (UML), which is primarily used in Object-Oriented -OO- software analysis and design, and the System Modeling Language (SysML). [27]
Organization	"Person or group of people and facilities with an arrangement of responsibilities, authorities, and relationships." [26]
Performance	"A quantitative measure characterizing a physical or functional attribute relating to the execution of a process,

	function, activity, or task."[26]
Point estimate	"The point estimate of a variable whose value is uncertain, is a single value for the variable in its range of possible values. From a mathematical perspective, the point estimate is simply one value among those that are feasible." [43]
Poisson Arrival Process	"The most important model for random arrivals is the Poisson arrival process. If A_n represents the interarrival time between customer *n-1* and customer *n* (A_1 is the actual arrival time of the first customer), then for a Poisson arrival process, A_n is exponentially distributed with a mean $1/\lambda$ time units. The arrival rate is λ customers per time unit. The number of arrivals in a time interval of length *t*, say *N(t)*, has the Poisson distribution with a mean λt customers." [77]
Process	"A set of activities, methods, practices, and transformations that people use to develop and maintain systems and associated products." [1]
Requirement	"A statement that identifies a system, product, or process characteristics or constraint, which is unambiguous, clear, unique, consistent, stand-alone (not grouped), and verifiable, and is deemed necessary for stakeholder acceptability." [26]
Sequence Diagram	"The sequence diagram represents specific interactions, commonly known as scenarios, among elements." [25]
Scenario	"Specific interaction among elements, characterized by a specific set of messages arriving among the modeled elements in a specific order." [25]
Singular	A square matrix with a null determinant. This type of matrix does not have a matrix inverse. It is also known as a non-regular matrix. Similarly, a non-singular matrix is a square matrix with a non-null determinant and has a matrix inverse (It is also known as a regular matrix). [36]
State	"A distinguishable, disjoint, orthogonal condition of existence of an element that persists for a significant period of time." [25]
State Machine Diagram	"A state machine diagram (STM) describes the state of a system or its parts; the transitions between the states; the actions that occur within states or upon transitions, entry or

	exit; and the events that trigger the transitions." [26]
System	"Broadly defined as an integrated set of elements that accomplish a defined objective. Every system is delineated by its spatial and temporal boundaries, surrounded and influenced by its environment." [6].
System life-cycle	"The evolution with time of an SOI from conception to retirement." [26]
System of Interest	"The system of interest to an observer." [1]
System of Systems	"Two or more systems that are separately defined but operate together to perform a common goal." [1]
System Thinking	"The process of understanding how those things which may be regarded as systems influence one another within a complete entity, or larger system." [6]
Type-II error	"The hypothesis is false, but it is accepted" [36] (a.k.a false-negative)
Verification	The process intended to "provide objective evidence that a system or system element fulfills its specified requirements and characteristics." [26]
Validation	The process intended to "provide objective evidence that the system, when is in use, fulfills its business or mission objective and stakeholder requirements, achieving its intended use in its intended operational environment." [26]

Chapter 1: Introduction

1.1 Background

This research results from a need to improve the understanding and management of complex systems from a holistic perspective. Stakeholders' expectations, globalization, market competitiveness, interconnectivity, and challenges at all levels of the global economy, environment, and society are key factors driving the rapid innovation and development of new, increasingly complex products. In recent decades, the growing complexity of engineering systems has been possible thanks to the accelerated progress of influential technologies in computers, electronics, communications, energy, material science, software (SW), chemistry, nanotechnology, and biotechnology. Developing highly complex systems and delivering sophisticated products under budget constraints, increasingly strict regulations, and market pressure is a challenge for companies, businesses, organizations, and engineers [45]. The challenges associated with system complexity require new ways to think about systems and new methods and tools to keep up with the fast pace of system complexity. Complexity is an essential system property inherently linked to what the system consists of and what it does. As the adage goes: if it cannot be measured, it cannot be managed. Therefore, measuring complexity allows system engineering professionals to know more about the system. Like any other property or phenomenon, measuring complexity is only conceivable with a tool and a measurement methodology. The better the tool and methodology, the better the measurement. In this research, an innovative measure is integrated with a model-based methodology to quantify complexity from a holistic perspective. The holistic approach observes the entire system as a whole, and the

relationships among its parts and collective functionality are considered as essential as the parts themselves. When the system components come together, their interaction originates function and system behavior, emergent properties that cannot be revealed simply by analysing the individual system elements. The integrated complexity measure and modeling methodology proposed in this research allow for the quantification and identification of complexity sources. The combined complexity quantification and knowledge extracted from the system model allow for inferences about the system's structure, functionality, and behavior. Structure, functionality, states, and behavior are system properties. By measuring these properties, the system engineer can identify the sources of complexity in the System of Interest (SOI). The identified sources of complexity can be a particular SOI's subsystems, components, elements, configuration, or functions. A better understanding of system complexity empowers system engineers and architects to allocate these sources of complexity and implement risk mitigation strategies that benefit the whole product life-cycle. The innovative complexity measure, integrated with a modeling approach and holistic vision, embodies a significant contribution to academia and industry. The proposed measure and methodology utilize a broad range of concepts from various fields, such as systems engineering, signal processing, information theory, network theory, and quantum mechanics. A more detailed explanation of these concepts is provided in Chapter 2.

1.2 Research Motivation

This work was motivated by the need to have a practical and holistic approach to quantify system complexity. System behavior is a crucial, but traditionally overlooked,

property of complex systems that needs to be explored further to better understand system complexity. Traditional approaches, discussed in Chapter 2, have primarily used a reductionist approach to understanding complex systems. In this type of approach, a system is dissected or atomized into its constituent parts, and the measure typically considers either system structure or function. However, a complex system as a whole is more than the sum of its elements. When all the system components interact as a whole, a new and essential system property emerges that is known as the system function: the purpose of the system [6], which cannot be explained solely based on an analysis of the system's parts. A sequence of functions and associated changes in the system state determine a system's behavior. As the structure or functions of a system become more interwoven, changes in either one can lead to changes in the system's behavior, which becomes more difficult to forecast as complexity increases. Thus, the level of difficulty in predicting the system's behavior is an indication of its complexity. Behavior is ultimately the reason for the system to exist, as it constitutes the solution to the customer's needs. Consequently, behavior is an essential system attribute that has to be included in quantifying system complexity.

Therefore, this work provides an alternative to the reductionist approach by implementing systems thinking, a relatively new concept aimed at analysing systems from a broad perspective and reducing blind spots. Systems thinking provides a comprehensive point of view to better understand a system. This work also incorporates Model-Based System Engineering (MBSE), a relatively new approach to facilitate complexity management, design quality, and communication among various stakeholders and system designers [27]. MBSE facilitates complexity management by enabling system

engineers to have more control, management, and accurate system representations of the system, which provides them with a better understanding of the system complexity. The integration of the proposed complexity measure with MBSE provides a solution to the systems engineering community for measuring system complexity with an objective and comprehensive complexity measure that incorporates structure, function, and behavior into a single quantifiable system property.

1.3 Problem Statement

System complexity has grown exponentially over the last three decades, but the development of complexity measures to quantify system complexity is still in its infancy. Most current complexity measures and method approaches overlook the development of highly complex system elements, which constitute the bulk of development effort, yielding inaccurate development times and program costs. Most complexity measures correlate system complexity only to either system's structural size or functional size. However, complexity measures based on system structure are sensitive to neither functional nor behavioral changes. System complexity needs to be defined in terms of structure or functionality size and behavior predictability [1][2]. System behavior has been particularly omitted in the literature, but it is a critical system attribute to consider when assessing modern systems' complexity [14]. System behavior is also strongly interrelated to system structure, function, and states. Measures that only include either structure or function and exclude behavior are limited in their ability to completely assess system complexity.

1.4 Research Questions

This research addressed the following questions:

1. Is system behavior a quantifiable magnitude, and how is it related to system complexity?
2. How can a holistic complexity measure be integrated with a model of a system?
3. Is a complexity measure based on a system's state and behavior approach more advantageous in assessing system complexity than traditional complexity measures?

1.5 Research Objectives

The following research objectives were set to address the above research questions:

1. Formulate a mathematical model of a complexity measure based on a system structure, function, and behavior approach.
2. Develop a system model using model-based methods and tools and integrate it with the novel comprehensive complexity measure.
3. Determine if a comprehensive complexity measure approach provides advantages over traditional approaches in measuring complexity.

1.6 Research Hypotheses

The following hypotheses were formulated to test the answers to the previously stated research questions:

1. System behavior is a quantifiable system property correlated to system complexity.
2. A resulting system model construct contains stored information that can be retrieved and quantified by a comprehensive complexity measure. This

hypothesis assumes that there is a link between the mathematical model and the system model. The system model contains model blocks that are presented in the diagrams. These blocks contain properties or parameters that are extracted to feed the mathematical model. Furthermore, the Mathematical model supports the analysis of the system's complexity based on the complex mathematical relationships that describe system behavior, function, and structure.

3. A complexity measure that accounts for system behavior, states, structure, and function is more sensitive to complexity changes than traditional complexity measures, which only account for structure or function.

1.7 Research Contributions

This work makes the following contributions to the systems engineering body of knowledge:

1. Formalizes system complexity as a single quantifiable system property of a complex engineering system.
2. Defines system behavior as a linear combination of basis functions representing a system state and scaled by the corresponding probability that the system is in that state.
3. Provides an objective and comprehensive complexity measure that associates structure, functions (with associated system state changes), and behavior.
4. Incorporates a modeling approach that facilitates a comprehensive description of system design, enabling higher-fidelity development

estimates.

1.8 Dissertation Outline

Chapter 1 presents background information related to system complexity, the relationship between structure, function, and behavior, and the role of all three elements in complexity quantification. The background is followed by the Research Motivation, Problem Statement, Research Questions, Research Objectives, and Research Contributions. Chapter 1 ends with this research outline.

In Chapter 2, several current complexity measures and their methodologies are analyzed. The review includes concepts necessary to understand the different literature approaches to quantify system complexity.

Chapter 3 includes the research methodologies used to answer each of the three formulated Research Questions, test the Research Hypotheses, and achieve the Research Objectives. The proposed complexity measure is obtained via a deductive mathematical derivation method. A model of an ATM system is developed, and the proposed measure is applied to the system model. The chapter finishes by analysing the advantages of applying the new complexity measure.

Chapter 4 contains two test cases using the system model developed in Chapter 3, where the new complexity measure is applied. Comparisons are made relative to a known complexity measure, and an analysis of the results is provided. After presenting the results, the chapter finishes by discussing whether the actual results validate the initial hypotheses.

Chapter 5 includes considerations and limitations of the research and determines if the problem statement and research objectives are fulfilled. Finally, the chapter highlights this research's contributions to systems engineering and the recommended directions for future research.

Chapter 2: Literature Review

This chapter presents an overview of a broad range of concepts from various fields, such as systems engineering, signal processing, information theory, network theory, quantum mechanics, as they relate to understanding complexity. A review of state-of-the-art complexity measures is also provided.

2.1 Systems and Complex Systems

A system is a set of correlated elements, components, or parts that interact to fulfill one or more intended purposes [26]. An engineered system, also known as an engineering system, is a human-made system designed or adapted to work in a known environment under certain conditions or constraints [1]. Engineered systems are also known more generally as products, with many forms that vary from hardware, such as computers, automobiles, aircraft, electronics, and machinery, to software (SW) [46]. Throughout this research, the general terms system and complex system refer to the more particular terms engineered system and complex engineered system, respectively. Complex engineered systems are engineered systems characterized by highly interdependent states and interactions that make their behavior difficult to predict. Since this research focuses on open systems, not on closed systems (those that do not associate with the environment [22]), the terms system and complex system are understood to include the system's environment or supporting infrastructure. Finally, when the system goes into production, it becomes a product.

The elements or components in a system constitute its structural basis, and their interaction within the system occurs by interconnections [15]. Interconnections are not

limited to their inclusion among the system's components, as some system components can establish interconnections with the system's external world or surrounding environment. How the components associate in the system is known as system configuration, which is a characteristic unique to the system, similar to a fingerprint. In a system, the components are associated with a purpose: the system function(s). When a stimulus or input is applied to the system, the system reacts by applying a function to it, transforming it into a response or an output. The typical representation of an open system is shown in Figure 1. In this representation, the system is a black box containing all its components [41]. The interface between the system and its surrounding environment is the system boundary (dotted line). The stimulus is how the environment interacts with the system, and the system response is the way the system interacts with the environment. How the architect defines the stimuli and response greatly influences the boundary and the system-environment relationship.

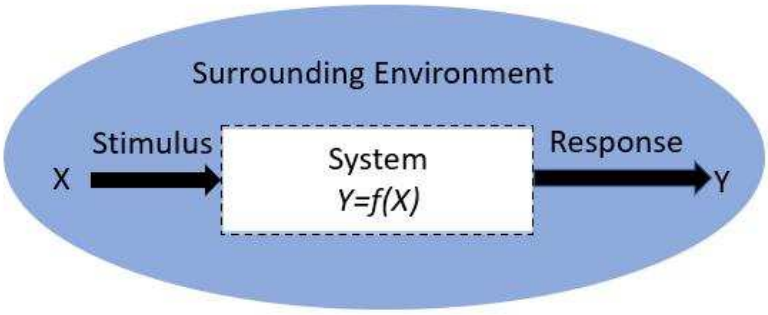

Figure 1. Classical representation of an open system.

The function *f* (or transfer function, as it is usually referred to in signal processing) establishes the relationship between an input *X* and output *Y*. The transfer function can be interpreted as an operation performed on the stimulus to obtain a result or output. The system's functionality rises beyond its structure. The primary function or

behavior of a system only emerges when its elements become organized and unified [15]. This emergent behavior (also known as the system's emergence) is the reason for its existence [26]. In complex systems, the system's emergence cannot be reduced to the components' behavior and is usually difficult to predict. This difficulty is partly because, in highly complex systems, the components can be themselves systems within the system (also known as subsystems) at several nested layers, each with its own structure and functionality.

2.1.1 System Structural Representation and Network Theory

Complex systems have numerous connections, not only among their components but also with the external environment. The number of connections typically tends to increase as a function of time [20]. Structure and function in a complex system are often elaborate, go hand-in-hand, and are influenced by each other. Inside the system of Figure 1 lies a structure of components associated with interconnections that form a network. Networks are the most common graphical description of a system. Conventional network notation uses nodes to represent system elements or components and links or arrows to represent the interconnections [60]. The composition of the network is also known as topology. Figure 2 illustrates four systems with different structures. The arrow's thickness indicates the relative strength (also known as weight) of the interconnections [59][1]. A thin link indicates a low or weak interconnection, while a thick one indicates a high or strong interconnection.

[1] In network theory, the relative strength of a connection between nodes dependes on the context and may refer to the frequency or amount of exchange (such as material, fluid, objects, information, or signal), or bandwidth, as in the case of computers in a computer network [59].

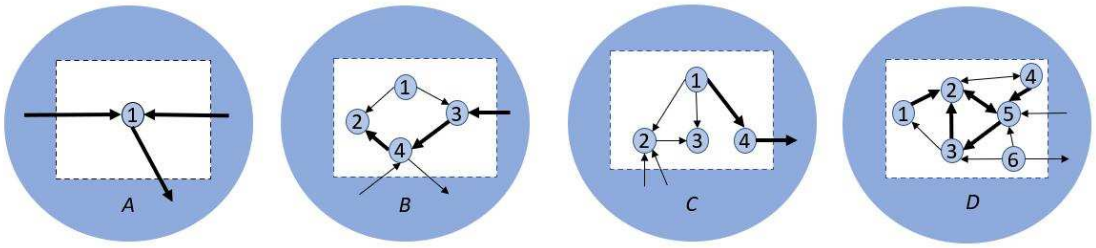

Figure 2. Systems with different structural complexity.

In Figure 2, the number of nodes suggests the initial criteria of a system's structural complexity. The network representation indicates that System *A* is structurally the simplest of the four systems because it has a single element. Note that System *A* presents three strong interconnections with its environment (blue area). System *D* is the most complex on the other end of the scale because it has six elements and 11 interconnections, the largest of all. Nine of System *D*'s interconnections are internal (five strong and four weak), and two are external (both weak), which indicates an overall weak interconnection with its surrounding environment. Interactions between components do not occur only in one direction. For instance, System *D* shows two bidirectional interconnections between nodes (a weak one between nodes 2 and 4 and a strong one between nodes 2 and 5). Systems *B* and *C* have the same number of elements (four) and interconnections (seven). These numbers indicate that Systems *B* and *C* have a higher structural complexity than System *A* and a lower structural complexity than System *D*. However, Systems *B* and *C* present different topologies. Internally, their elements connect differently and at different strength levels. System A has a ring-type topology with two strong and two weak connections, while System *C* has a branch-like topology with three weak interconnections and one strong interconnection. Externally, Systems *B* and *C* have the same number of interconnections with the environment as

System *A*, but only one interconnection is strong. The number and directionality of the interconnections are not the only indications of interconnection complexity. Other interconnection descriptors, such as color, bandwidth, communication protocol, and type of exchange at the interconnection (e.g., information, energy, or matter), can also provide meaningful details about the type of connection [16][17].

2.1.2 Design Structured Matrix and Adjacency Matrix

The Design Structured Matrix (DSM) is a visual and concise modeling tool used to represent the components and their interactions in a system's structural network [46]. The DSM is a square matrix of size N x N (N rows by N columns) mapping the interactions among the array of N system's components. The creation of a system's DSM is comprised of three fundamental steps:

1. Decompose the system into its components.
2. Identify the relationships between components.
3. Display the components and their relationships using a geometric, alphabetical code or numeric notation.

The adjacency matrix is a type of DSM that uses a binary numeric notation, placing "1" to indicate a relationship between components and "0" elsewhere. Mathematically, the matrix elements $a_{i,j}$ represent the interaction (or link) between the system elements (or nodes) *i* and *j* [18]. If there is an interaction between the elements *i* and *j*, then $a_{i,j}=1$, otherwise $a_{i,j}=0$ (including elements' self-interactions, denoted as: $a_{i,i}=0$). Conventionally, the order of the subscripts indicates the direction of the interaction. For instance, $a_{i,j}=1$ represents a unidirectional link from component *i* to

component *j* (furthermore, $a_{j,i}=0$). A bidirectional link between component *i* and component *j* is represented as $a_{i,j}=1=a_{j,i}$. A representation of System D (see Figure 2) as an adjacency matrix is shown in (1) [47]:

$$A=(a_{i,j})=\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}=\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \quad (1)$$

2.1.3 Requirements, system adaptation, and system behavior

The response of a complex system may vary as a result of different input conditions. This characteristic is known as an adaptation [60]. In engineered systems, new inputs are equivalent to new or modified system requirements implemented as system redesigns by the system architect during the modeling stage and subsequently materialized by a cross-disciplinary group of engineers during the product development stage [26]. In other types of complex systems, such as self-adaptive systems (or self-organizing systems), the system reorganizes itself to adapt to the new environmental conditions [19]. Adaptation as a function of time is commonly referred to as evolution [60]. Adaptation requires functional changes, but not necessarily structural changes, as demonstrated later in Chapter 4. However, every time the system's structure changes, new functionality and system behaviors arise. A certain level of behavior unpredictability characterizes highly evolved systems. If the system behavior is too difficult to predict, identifying behavioral patterns known as fractals is more feasible than pursuing an exact prediction of the response [60]. Complex engineering systems are the main focus of this research. Thus, self-adaptive systems are out of scope.

2.2 Life-Cycle

A complex system is, after all, a system, and as such, it experiences a series of phases that collectively are called product life-cycle. A product life-cycle is a series of phases through which a system or a manufactured product is subject to specific activities or processes from the initial concept to utilization and final disposal [26]. These activities and processes are necessary to obtain an outcome or product that satisfies the initial customer requirements. Organizations usually establish enter and exit criteria at each life-cycle phase to control the transition from one phase to the next. Thus, milestones and gates are implemented as project management decision points to determine whether the project can continue to the next phase.

Organizations choose one of several life-cycle approaches used in the industry, depending on the company's profile, business mission, business type, process definitions, and organizational idiosyncrasies (at the corporate, team, and individual levels). These dependencies also affect the product's complexity. A generic life-cycle includes the following phases, each of which is composed of a series of activities [48]:

1. Concept: This phase is characterized by the identification of customer needs, exploration of feasible concepts, and selection of viable solutions. The preliminary concept is also used to generate an early estimation of the project's cost and schedule.

2. Development: In this phase, the system requirements and constraints are defined, characterized, and refined. At this point, the system's requirements are interpreted, and the system's emergent function is identified [29]. This phase also includes the creation of an initial architecture and design of the

system. Input from stakeholders inside and outside the company is typically incorporated through planned technical design reviews and decision gates. In this phase, detailed verification, validation, and integration plans are developed and executed. Finally, the estimation of cost and schedule is refined and detailed. Having a good understanding of the system complexity at this stage is an invaluable asset that yields more accurate project predictions. In a complex system, the development phase is ideal for performing any design changes. Design changes performed later in the product life-cycle are usually both more challenging and more expensive to implement.

3. Production: In this phase, the system design is implemented, inspected, and verified for manufacturing or production.

4. Utilization: In this phase, the product is operated in its intended environment to perform its intended function. As a result of the product's utilization, some opportunities to enhance the product can be identified, and design changes may be made.

5. Support: This phase consists of product maintenance and service. The former activity in this phase ensures the product stays in operational condition, and the latter corrects any malfunctions. Information collected during this phase can also lead to design changes.

6. Retirement: This is the final phase of the product life-cycle and includes storage, archival, recycling, refurbishing, or disposition.

Typically, engineering activities occur in the concept and development phases (also known as engineering phases). Re-engineering occurs in the production, utilization, support, and retirement phases (also known as re-engineering phases). After a feasible system solution is conceptualized in the concept phase, the system definition, verification and validation (V&V) plans, and V&V execution have to be carefully and sufficiently detailed in the development phase. The design definition is critical for any system, but particularly for complex systems. In the design definition activities or design architecture, all possible aspects of the system design and associated technical and business risks must be considered. Understanding, controlling, and predicting all these aspects and risks becomes an immense engineering challenge in a complex system. Unsurprisingly, the development phase is often susceptible to design errors and omissions, incomplete V&V testing, incorrect conclusions, and inaccurate project estimates [6]. In addition to activities and processes, the product life-cycle phases involve physical resources, people, schedules, and organizations across a company. Subsequent phases (production, utilization, support, and retirement) are significantly affected by architectural decisions originated in the development phase. Fixing errors in the product life-cycle re-engineering phases is more challenging and expensive than if the errors are caught and corrected in the concept or development phases [61]. The quantification of system complexity during the design phase helps systems engineers identify and analyse complexity sources. Based on this understanding, systems engineers can draw conclusions, correct the system design, adopt risk mitigation strategies, and refine project estimates [6].

2.3 System Architecture Process

Creating a complex system architecture is challenging because it usually has plenty of design parameters whose combinations produce a vast solution space. The principal goal of the system architect in managing complexity is to minimize ambiguity and uncertainty. To achieve that goal as much as possible, the architect must be rigorous and systematic in approaching the architectural process and select the right toolset to construct and view the architecture. As stated in the previous section, the architecture activity is an integral part of the development phase. The system engineer or architect defines the technical system requirements as a first step, translating the stakeholders' needs and desires into technical solutions that satisfy requirements from a user perspective [48]. In this step, the architect identifies the system's emergent property to derive the necessary functions, internal constraints, and environmental constraints. Next, the architect generates an engineering solution to the customer needs within certain limitations and conditions [47].

The system's architecture is usually described using architectural frameworks, which are standardized viewpoints and models that make it easier to communicate the architecture and address diverse stakeholders' concerns [26]. Modeling and simulation techniques in the engineering phases are instrumental in describing complex systems' architecture. These techniques can provide information that facilitates design and architectural decisions before valuable resources are spent later in the product life-cycle. System modeling is intended to capture the system architecture and design, making it easier to understand certain phenomena, functions, or processes, and identify performance limitations [26]. The term simulation is often interchanged with modeling,

but more strictly, simulation refers to the implementation of a model under particular scenarios and conditions [6].

2.4 Models and Model-Based System Engineering

The complexity measure developed in this research is applied to a system model. The MBSE approach is followed to facilitate the system modeling process. A particular application of MBSE is illustrated in Chapter 4, where test cases exemplify its use. Thus, this section's remainder is dedicated to reviewing complexity-relevant concepts related to models, modeling, and MBSE, which are applied in this research.

A model represents a real-world system or SOI and the interactions with its surrounding environment in a simplified and abstract manner [21][22]. Models serve various purposes at the development stage. Various agents, such as stakeholders, architects, and specialty engineers, may have different interests and, consequently, need different model perspectives. However, the information contained in the system's model is the point of confluence for all these agents to understand the system and communicate among themselves based on common ground. Views are usually generated from models and are intended to provide different perspectives to address the agents' varying interests and facilitate their decision-making processes [26].

Models can be classified in many ways. Most models may fall into one of the following broad categories [22]:

1 Physical models. These models are versions of an SOI at a particular scale and are typically used for demonstration, simulation, and prediction. Examples of these models include the solar system, atomic structures, aeronautic

assemblies, and industrial plants.

2 Abstract models. These models represent processes, phenomena, and physical or abstract systems, such as a financial system, using specific modeling constructs. Nowadays, these models are typically implemented using computers and SW programs written in a particular programming language. There are two main categories of abstract models:

   a. Analog models. These models exploit similarities between known models from various fields or disciplines to describe the SOI. For example, electric circuit models can describe mechanical, hydraulic, economic, or biological systems.

   b. Mathematical models. In this type of model, the SOI is described in mathematical symbols and formulations. These models provide a high level of abstraction and precision and can allow for the incorporation of probability to account for random factors inherent to most real-world systems.

Traditionally, systems engineering has considered documents the primary work artifact, such as system requirements, specifications, structure, functions, interfaces, and behavior, supporting all or part of the system design and development life-cycle activities. MBSE is a model-centric system engineering approach that considers models the source and the ‘house” of information. Documents are considered data repositories and a subproduct of the integrated model. Most MBSE applications use a graphical language called Systems Modeling Language (SysML) to communicate the system’s

model [27][28]. Like any other communication language, SysML consists of symbols, syntax, and semantics. SysML is mainly based on the Universal Modeling Language (UML), the Object Management Group standard visual language. In terms of model elements, a class in UML is equivalent to a block in SysML, class associations are block associations, class attributes are named block values, and so forth.

Similar to UML, SysML implements two basic diagram categories: the behavior diagrams and the structure diagrams. In addition to the structure and function, the behavior is an essential system attribute that denotes the change in system state as a function of time [25]. SysML recognizes the importance of these two attributes and preserves that legacy from UML. SysML also introduces two additional types of diagrams, the requirements and parametric diagrams. SysML uses four essential behavioral diagrams:

1. Activity diagram: Represents the flow of information (data, material, or energy) and control between connection points (activity parameters).

2. Sequence diagram: Represents the interactions (also known as scenarios) among system components and components with external elements that occur in a particular order or sequence.

3. Use case diagram: Represents collective functionality accomplished among system components and may include relationships with external components or systems (e.g., users).

4. State Machine diagram: A state is defined as a system's existential condition. This diagram describes the system state transitions and can include actions or events

that originate the transition and consequent system reaction or response.

SysML also defines four structural diagrams:

1. Package diagrams: Represents a large partition of the system, grouping the central system's components into what are known as packages (or sets of blocks) in an organized manner.
2. Block Definition Diagram (BDD): Equivalent to the UML's class diagram, this diagram represents system components and their relationships. They also can depict other block properties, such as associations, values, services, and constraints.
3. Internal Block Diagram (IBD): Equivalent to the UML's structure diagram, this diagram represents the elements and their relationships, which form the block's internal structure.
4. Parametric diagram: One of the brand-new additions in SysML, which enables the system's constraints description through equations or inequalities. Because this diagram establishes a bridge between the system's design model with engineering analysis models, it is advantageous to use in deriving computational algorithms and calculating system properties.

The requirement diagram is a new type of diagram in SysML that does not fall in either the behavior or structure diagrams. However, the requirement diagram is a unique graphical construct in SysML to capture the system's requirements, requirement relationships, and requirement hierarchy. Because this diagram establishes a bridge

between the system's design model and requirement management tools, it becomes beneficial for traceability and verification purposes. The requirement traceability and verification become critical activities because requirements are what triggers complexity changes in a system. In complex systems, traceability becomes challenging without an automated modeling tool.

2.5 The Object-Oriented Systems Engineering Method

Comprehending and modeling complex systems involves a diversity of disciplines that must be unified in a common framework, which is necessary for developing and communicating the system model regardless of the type of system, domain, and level of complexity. The Object-Oriented Systems Engineering Method (OOSEM) is the underpinning framework of MBSE that integrates object-oriented and systems engineering concepts to facilitate the system's analysis, specification, design, and verification. Following the OOSEM method systematically becomes particularly helpful for the architect in reducing ambiguity and uncertainty during the architecting and development of a complex system. The methodology consists of the following activities [26]:

1. Analyse stakeholder needs: This activity consists of understanding the customer's problem, setting the system's goals, and defining the effectiveness measures to evaluate if the system satisfies the customer's needs and meets the goals.

2. Define system requirements: This activity determines the required functionality, interfaces, constraints, physical characteristics, and desired system performance to support the effectiveness measures and achieve the goals.

3. Define logical architecture: This activity is part of the system architecture design. It decomposes the system into logical components and creates logical scenarios where the logical components interact to perform each system's function or operation.

4. Synthesize candidate physical architecture: This activity determines the physical components, relationships, and correspondence with the logical components. It also establishes traceability between requirements, functionality, and physical components.

5. Optimize and validate alternatives: Different system designs are optimized in this activity, and multiple aspects such as performance, reliability, serviceability, and usability are evaluated. Then the different designs are compared, and a design is chosen based on specific criteria.

6. Validate and verify system: In this activity, the system is evaluated to determine if it meets requirements and validate that those requirements meet the customer needs.

2.6 Information Theory

Signal processing principles are one of the pillars used to develop the proposed complexity measure in this research. However, undergirding these principles are essential concepts of information theory, such as source coding for information transfer (from a source to a recipient using a communication channel), information measurement, and the channel's information capacity [49]. Many complexity measures found in the literature are based on information theory [7][11][63]. Central to information theory is the

information measure. Shannon defines the information measure as $I_i = log_b(P_i) = -log_b\left(\frac{1}{P_i}\right)$, where I_i is the self-information of an arbitrary message x_i, $P_i=P(x_i)$ is the probability of the event that the message x_i is selected for transmission (uncertainty), and b is the logarithmic base, which determines the information unit [49]. The likelihood of a message occurrence is inversely proportional to its information content. In other words, the information measure is related to uncertainty. To illustrate this concept, suppose that an individual plays a game consisting of guessing a number between 1 and 1000 (i.e., the game has 1000 possible outcomes). At the beginning of the guessing game, the player's chances of guessing right are 1/1000 (or 0.1 percent chance). However, the individual can answer with 100 percent certainty after asking only 10 yes/no questions (i.e., is the number greater than 500? If it not, is it greater than 250?…and so forth). Each yes/no answer provides a bit of information to the player, and with each bit of information, the number of possible outcomes is halved. Then, in each of the ten rounds of questions, the information measure is reduced by 10 percent.

Binary code can be used to represent a bit with just two binary numbers: 0 and 1. Then, each answer (or instruction or symbol) can be encoded with a bit string assignment; a “no” corresponds to the digit 0 and “yes” to the digit 1. Then, to answer with 100 percent certainty, the player needs ten bits of information arranged in a string of ten binary numbers. Shannon’s equation predicts that a question of N possible outcomes can be answered with a string of only $log_b(N)$ bits of information, when a bit of information consists of b symbols. In the example, if the number of possible outcomes is N=1000, then discerning among these 1000 possibilities requires $-log_2(1/1000)$=10 bits of information, when a bit consists of two symbols: 0 and 1.

Information can also be interpreted as a measure of order and disorder or simplicity and complexity. A distribution function is a statistical description to deal with the uncertainty associated with the probability P_i. Entropy is another term closely tied to information. Shannon derived his mathematical expression of information based on Boltzmann's second law of thermodynamics [64]. Entropy is an essential property of a collection of objects or a system that describes its most probable configuration (also known as microstate) or outcome (also known as macrostate). Boltzmann entropy is defined as $S = k * log_b W$, where S is the entropy of a system, k is the Boltzmann's constant, b is the logarithmic base, and W is the number of possible microstates that correspond to the macroscopic thermodynamic state. In the Boltzmann equation, the entropy is equivalent to the amount of information needed to describe the system's microscopic state, assuming its macroscopic description is known. An increase in the system's entropy always originates a loss of information and vice versa. The second part of this chapter includes several structural complexity measures from the literature, which were developed based on the information theory concepts reviewed here.

2.7 Molecular Quantum Mechanics Theory Overview

The review of Molecular Quantum Mechanics (MQM) in this section is necessary to understand the principles and assumptions that gave origin to the current state-of-the-art system complexity measures.

The origin of MQM is associated with network theory. Network theory (discussed in Section 2.1.1) provides the theoretical and descriptive toolset to represent a system's structure. Eugene Wigner pioneered the application of networks to depict physical systems [28]. Wigner demonstrated in the 1930s that the energy spectra and other similar

atomic nuclei features could be described in terms of eigenvalues of random matrices. Also in the 1930s, the solution of the time-independent Schrodinger equation with approximations was found by Erich Hückel. As part of his solution, Hückel proposed a method, the Hückel Molecular Orbital method (also known as HMO theory), to predict the energy and behavior of a particular class of polyatomic hydrocarbon molecules (known as conjugated π-systems) [31]. Hückel's theory is considered the basis of Graph Theory [32][33]. In the HMO, the hydrocarbon molecules are described using graphs. A graph's vertices (or nodes) represent the carbon and hydrogen atoms, and the graph's edges (or links) between vertices represent the bonds between carbon atoms and carbon and hydrogen atoms. The graph's mathematical representation is the matrix used to solve Schrödinger's eigenvalue/eigenvector equation. The molecule's total π-electron energy is then found as the sum of the matrix eigenvalues' absolute values.

The Schrödinger equation is one of the most fundamental physics equations and involves a vital concept, the wave function. The wave function ψ is a function that contains all the information about the state of a physical system at a particular time [37]. The term wave function and state function may be used interchangeably. The general time-independent Schrödinger equation (also known as the eigenvalue equation or eigenequation) is a linear differential equation that characterizes the wave function of a quantum-mechanical system:

$$H\psi = \xi\psi \tag{2}$$

where ψ is the wave function or state function, H is the Hamiltonian operator, and ξ is the corresponding eigenvalue(s) of H [34][35]. In the HMO theory, the molecular orbitals of

the network of atoms in the π-system are represented with the wave function ψ expressed as a linear combination of n normalized basis functions ψ_i:

$$\psi = \sum_{n} c_i \psi_i \tag{3}$$

where ψ represents the molecular orbitals of the atoms' network in the π-system, c_i are scaling coefficients, ψ_i are normalized basis functions, and i is a subscript varying between zero and the maximum number of normalized basis functions n.

The optimization of the partial derivative of the eigenvalues ξ with respect to c $\left(\frac{\partial \xi}{\partial c} = 0\right)$ in a simultaneous equation system allows for the determination of the individual coefficients c_i:

$$\sum_{j=1}^{n} c_j \left(H_{i,j} - \xi I\right) = 0 \quad (i = 1, \ldots n) \tag{4}$$

where the Identity matrix I and the Hamiltonian H are matrices of size $n \times n$. Non-trivial values of the energies ξ in (3) are found by solving the secular determinant with Cramer's rule [36]:

$$det\left(\left|H_{i,j} - \xi I\right|\right) = 0 \tag{5}$$

where H is the Hamiltonian operator in matrix form, ξ is the vector of the corresponding eigenvalue(s) of H, and I is the identity matrix. Introducing some simplifications in the Hamiltonian matrix allows for the identification of an approximate solution to the Schrödinger equation of an unsaturated hydrocarbon, as shown in the following parametric equation of the Hamiltonian matrix:

$$H_{i,j} = \begin{cases} \alpha, & i = j; \\ \beta, & i,j\ adjacent; \\ 0, & else \end{cases} \tag{6}$$

where $H_{i,i}=\alpha$ is an isolated electron's energy, and $H_{i,j}=\beta$ are the energy interactions between orbitals on adjacent carbons. $H_{i,j}=0$ represent null interactions between non-adjacent carbons. Then, the Hamiltonian system's energy representing the organic molecule network is contained in the adjacency matrix. In other words, the molecular orbital energy levels of the π-electrons in hydrocarbon molecules coincide with the molecule graph's eigenvectors [31]:

$$\xi_\pi = \sum_{i=1}^{n} h_i \xi_i = n\alpha + \beta \sum_{i=1}^{n} h_i \sigma_i = n\alpha + \beta \left(\sum_{i=1}^{n} h_i\right)\left(\sum_{i=1}^{n} \sigma_i\right) \tag{7}$$

where ξ_π is the total energy of all π electrons, h_i are weigh factors for each energy level, α and β are constants, n is the number of carbon atoms in the molecule, and σ_i are the singular values of the adjacency matrix $\sum_{i=1}^{n} \sigma_i$.

2.8 System Complexity Definition

One of the earliest studies on complexity defines a system's complexity in terms of system characteristics (such as the number of constituents and their corresponding interconnections) and by the level of difficulty in predicting system behavior [3]. More recent literature provides two additional concepts to the previous definition: (a) the relationship between the system and (b) its environment and the emergent purpose of the system as a whole [6]. The former concept suggests that real-world systems, which are generally open systems, do not exist in isolation. Their interactions with the surrounding environment directly or indirectly influence their complexity. The latter concept indicates that the system's functionality is not merely obtained by adding components or the sum of the individual entities' functions [15]. Measures to quantify the complexity of systems

are abundant in the literature. However, most of them were created for specific fields or for solving particular problems [17]. Also, complexity measures are calculated based on various criteria, including information, entropy, topology, cost, structural size, and functional organization, to name a few [4]. The remainder of this chapter discusses relevant measures and methodologies proposed in the literature to assess system complexity. The concepts necessary to understand and analyze the various complexity measures and associated methodological approaches are also included.

2.9 Static and Dynamic Complexity Measures

A dynamic complexity measure based on information theory and entropy state probability analysis was created to find the optimum time to reconfigure manufacturing systems [7]. The complexity measure is classified as dynamic because it considers a production system's reconfigurations in different variable length periods. Industrial manufacturing plants such as cars and electronics must be dynamically reorganized for frequent product launches and updates. The reconfigurable manufacturing system state depends on the interaction between processing function, processing capability, and product demand. The dynamic complexity of manufacturing systems does not respond to a stable schedule plan due to factors of uncertainty such as product demand. Therefore, the dynamic complexity of manufacturing systems correlates with the variability of processing function and processing capability. The dynamic complexity measure was developed using a mathematical derivation methodology to obtain an expression similar to Shannon's information equation [49]. The dynamic complexity measure assesses the system's entropy as a weighted sum of a positive and a negative complexity. The positive complexity term represents the system's entropy in a normal operation state (stable

system or non-blocked state). The probability of an event's occurrence in Shannon's equation is substituted by a multidimensional array containing the non-blocked state's probabilities for each part, cell, and processing function. The negative complexity term is the sum of the system's entropies in an efficiency declining system state (system with compromised stability or blocked state) and non-operational states. In this case, the probability of an event's occurrence is replaced with the sum of two multidimensional arrays. The first array contains the blocked state's probabilities, and the second array contains the non-operative state's probabilities. Both arrays are evaluated for each part, cell, and processing function. Based on the entropy calculation, the state probabilities are converted into a system's complexity quantification. The verification methodology of the dynamic complexity measure is empirical. Based on historical data from a manufacturing plant, parameters such as order change and machine tool failure allow the calculation of the processing function and processing capability states. Failure to satisfy the demand for particular parameter combinations originates system instability, reflected in an entropy increase and higher values of the dynamic system complexity. The maximum dynamic complexity value serves to identify the critical state at which the manufacturing system should be reconfigured. In contrast to this dynamic aspect of manufacturing, there are static periods between plant reconfigurations, where the previous complexity measure is not applicable because the system is stable.

Another complexity measure, also based on information theory, covers the static complexity characteristic of manufacturing systems [5]. Even though there is no dynamism in this situation, the manufacturing process or systems may still be complex. The static complexity is defined in terms of factors such as the part types produced in a

single run, the multiple operations, machines or processors, and the constraints necessary in each operation to produce each part. The static complexity measure accounts for a system's structure consisting of machines or processors, parts, and interactions. This structural information is obtained from historical production orders and process plans detailing individual parts' production. The static complexity measure's derivation uses a mathematical methodology to obtain an expression that resembles Shannon's information equation [49]. The static complexity measure quantifies the system's information by applying the probabilities of multiple events represented in a high dimensional array that contains the processing time for each operation, machine, and part.

Unfortunately, the dynamic and static complexity measures reviewed above are only applicable to manufacturing systems. As seen in the case above, two measures are needed to cover a manufacturing system's two complexity aspects. Measure specialization is a shortcoming of many existing complexity measures because it limits the measure's transferability. In some cases, two or more complexity measures are often needed to assess various aspects of a system's complexity. Most complexity measures are so specialized that they are not transferrable to other domains, industries, applications, processes, product life-cycle phases, or scenarios.

2.10 Engineering Design and Complexity Measures

Engineering design is the process of translating customer needs and desires into design solutions. Several complexity measurements in the literature have been proposed to evaluate the complexity of any of the three aspects involved in engineering design: design problem, design process, and design products [2] [13].

2.10.1 The Design Problem

The critical processes of the design problem are defining the system requirements and analyzing the design concept. These processes are typically iterative as requirements are refined continuously and more information is assimilated to progressively mature the design concept. The purpose of reiterating these processes is to match various functions and requirements. This matching process is typically not a trivial proposition in a complex system because requirements can have multiple attributes and associated impacts on the design. Finding the relationships between the design functions (variables) and the design's behavioral properties (attributes) under certain constraints such as customer objectives is a challenging optimization problem. One way to reduce the optimization problem's scale is by decomposition analysis, which converts the extensive optimization problem into several smaller and more manageable optimization problems. The decomposition implements several strategies based on DSM, such as tree-dependency analysis, matrix partitioning, clustering, and block optimization [68]. The use of DSM makes this approach firmly focused on structure complexity, and behavior and functionality are not considered. It also relies heavily upon reductionism. The reductionist approach assumes that a system is decomposable and that the whole system can be understood by dissecting and analyzing its parts. This assumption is not realistic for complex systems in which functionality and relationships between parts become very relevant. In complex systems, the analysis of individual parts neither explains what all the parts do when coming together nor the whole system's behavior [20].

2.10.2 The Design Process

The next aspect involved in engineering design is the design process,

characterized by two main processes, the architecture and the design definitions [67]. On one side, the architecture definition consists of the activities necessary to create a general design solution to address the stakeholders' concerns based on the SOI's features, architectural principles, and concepts. On the other side, the design definition complements the architecture definition by adding a more detailed analysis of system performance and viability, creating a more concrete version of the architectural design based on the SOI's physical and tangible properties, such as structure, operation, and environment constraints [65]. The design process itself can be understood as a particular class of system. Process entities such as activities, tasks, resources, organizational units, events, and other artifacts can be considered components of a graph-like structure. Once the graph assembles the entities and their interrelationships, the graphs' structural complexity can be evaluated with central tendency-type statistical measures [66]. The design process' complexity analysis is a goal-oriented procedure pursuing graph pattern optimization to improve workflow. However, optimizing these procedures may become a very intricate proposition. It is possible to use nonlinear autoregressive neural networks in complex systems to facilitate pattern identification and performance prediction [66].

2.10.3 The Design Product

The design product consists of the activities necessary to develop the design solution selected in the design process [19]. The design product processes include system analysis, system development, system integration, and V&V. The systems analysis objective is to understand the design technology and provide project estimations that enable stakeholders to make informed design and business decisions throughout the product life-cycle. The system development process implements the design according to

the requirements, architecture, design, and constraints. The system integration process consolidates the subsystems or elements into a realized system focusing on the interfaces to ensure that it operates as intended. Finally, the integrated system is tested during the V&V process to provide objective evidence that the system implementation (i.e., product) meets specifications and the user's operational needs.

2.10.3.1 A Design Product Complexity Measure Based on Function

A parametric equation that relates the effort associated with a design product as a function of product complexity and difficulty level to meet the requirements is provided in [9]. The equation was determined empirically by regression analysis of historical data of a set of projects for two companies. Typical industry practice is to apply subjective measures to assess product complexity. This practice is not free of a certain degree of inaccuracy and risks. As a solution, [9] proposes a product complexity measure related to the consumption of resources. Despite acknowledging that structure and functionality are essential complexity features, it is argued that quantifying system complexity based on the number of components and interrelations (structure size) may be misleading because the complexity may be concentrated in just a few components. Therefore, measuring the number of functions (functionality size) is a more accurate quantifier of development effort because function and effort are: 1) directly correlated, 2) based on requirements, 3) independent of the product design method applied, and 4) a common ground concept in multiple disciplines. However, because some functions are more complex than others, measuring only the functionality size could lead to an insensitive and inconsistent complexity assessment: insensitive because significant complexity arising just in the structure can be missed, and inconsistent because the complexity of the holistic system

should be greater than that of the sum of the individual parts. A function decomposition and hierarchy methodology are proposed in [9] to remedy the inconsistency issue. Then, the product complexity measure is calculated as the sum of the number of functions at each level of the hierarchy decomposition. A correlation between product complexity and design effort is subsequently found using regression analysis. The correlation is validated based on empirical data from several projects with known figures for product development efforts from two companies. An advantage of this measure is its relative simplicity and ease of implementation. However, the accuracy in estimating development effort depends heavily on data accuracy from previous projects. An even more critical factor in assessing complexity more realistically is that the proposed function decomposition is, to some extent, a subjective methodology that varies according to the designer's discretion. And unlike most complexity measures, the proposed complexity measure in [9] includes functionality, but unfortunately, it overlooks structure. Structure and function are two essential complexity attributes of the system. There is a crucial bidirectional relationship between them, even though it is not necessarily linear. Small changes in structure can lead to a more sophisticated functionality and vice-versa. For instance, suppose that it is desired to add a single but essential function to a system. This functional change likely requires substantial structural changes too. A complexity measure based only on functionality would be insensitive to the substantial structural change and would likely report a small or neglectable complexity increment.

2.10.4 Remarks on Engineering Design Complexity Measures

As seen above, engineering design complexity has been firmly focused either on structural complexity (number of components, interrelationships, and configuration) or

functional complexity [2] [9] [13] [19] [20] [65] [66] [68]. However, structure and function are not mutually exclusive in complex systems and typically evolve together. They become so intertwined and mutually influential that even small changes in either one may significantly impact state changes and, consequently, system behavior. Although models are used in some of the measures and methodologies to quantify engineering design complexity, complexity is found indirectly by optimizing DSM models, assessing performance, or using subjective methodologies. Many other complexity measures are not general enough to apply to other complex systems, are insensitive to complexity changes, or are not easy to implement in real-world applications. In many cases, the measures' theoretical and practical limitations make them unsuitable for real-world systems. In summary, there is virtually no literature available on objective complexity measures that consider the system as a whole and can be applied directly to the system's models.

2.10.4.1 Complexity Measures and Dynamicity

An additional dimension of complexity is time. With the constant refining and addition of system requirements, components, interactions, and functionality, systems become increasingly complex over time. From this perspective, complex systems exhibit a changing (or dynamic) behavior as a function of time, which is progressively more challenging to predict. A measure based on the Suh Information Axiom [13] and the Goal, Question, Metric Approach [58] has been proposed to measure dynamic complexity [12]. The paper asserts that most static and dynamic complexity measures overlook historical project information, which is essential information to predict future system behavior. However, the complexity measure in [12] relies heavily upon Subject

Matter Expert (SME) knowledge. SMEs' opinions are usually non-transferable to other systems because of the degree of associated subjectivity, which can lead to bias, inconsistent results, and unsuitability for systems comparison purposes. Similarly to [7], the complexity measure is based only on functionality, and structural complexity is not considered.

2.11 System Structural and Functional Complexity Measures

2.11.1 A Disjointed Structural and Functional Complexity Measure

One of the first works on complexity measures identified structural complexity and functional complexity as two fundamental aspects of design complexity [8]. Design structural complexity was defined in terms of quantifiable information content in the design's internal structure. Similarly, design functional complexity was defined in terms of quantifiable information content in the design's functions. Since information content is the amount of information conveyed with the minimum possible number of binary bits, the best design process has the minimum information content that, according to information theory, can be quantified in terms of entropy (see Section 2.6)[2]. On one hand, the design process's structural complexity is related to the interfaces' information content. On the other hand, functional complexity is related to the information content associated with the probability of satisfying the given functional requirements. The structural and functional complexity measures in [8] were obtained based on a deductive mathematical

[2] Entropy is a measure of uncertainty. The term is used here in the sense of Shannon entropy, rather than the more general Boltzmann definition of entropy. Shannon defines the information content of a message communicated from a source to a receiver in terms of entropy, where the message can take different forms, such as a word, a sentence, a paragraph, a report, or a bit (a *0* or a *1*). The more information is obtained, the more uncertainty is removed, which indicates less entropy in the source.

derivation. The derivation defines the design process as a function of information content and takes advantage of the mathematical analogy between the design process and thermodynamics.

One of the main advantages of the proposed dual measures in [8] is that they enable designers to calculate a design's complexity artifact or design process. However, one of the disadvantages is that the measures quantify structure and function independently as if they were two separate entities of the same system. The measures' application is reductionistic rather than comprehensive, and neither system states nor behavior is considered. Another critical shortcoming is that the measures are intended for a constructed design, and no modeling methodology is proposed. Thermodynamics principles require the pre-existence of a design before evaluating its entropy performance. Finally, no examples or validation of the disjointed structural and functional complexity measures are provided.

2.11.2 A Molecular Quantum Mechanics-Based Structural Complexity Measure

One of the most recent measures proposed in the literature quantifies an engineered system's complexity C as the sum of two structural complexities terms [10]:

$$C = C_1 + C_2 C_3 \tag{8}$$

The first complexity term C_1 relates to the individual components' complexities. The second term, C_2C_3 involves combining the interface design activity C_2 and the system's connectivity structure (topology) C_3. C, C_1, C_2, and C_3 are unitless scalar

quantities. The complexity measure in [10] was derived following a deductive and mathematical approach. The derived mathematical model applies graph theory principles (see structural representation in section 2.1.1), exploiting the structural similarity between engineering systems and some hydrocarbon molecules, called polyatomic conjugated π-systems. One of the simplest examples of a conjugated π-system is the Benzene molecule (also known as Ethylene: H_6C_6). The hydrocarbon molecule of benzene consists of six carbon atoms and six hydrogen atoms. Each carbon atom is connected to a hydrogen atom and a carbon atom with a single bond (σ-bond) and another carbon atom with a double bond (π-bond). Therefore, every two carbons overlap axially to form a double bond (π bond). In Figure 3, the benzene's molecular formula is shown on the left, and its network model is shown on the right. In the model, the set of carbon and hydrogen atoms are represented as nodes (balls), and the bonds or interrelations are represented by links (sticks).

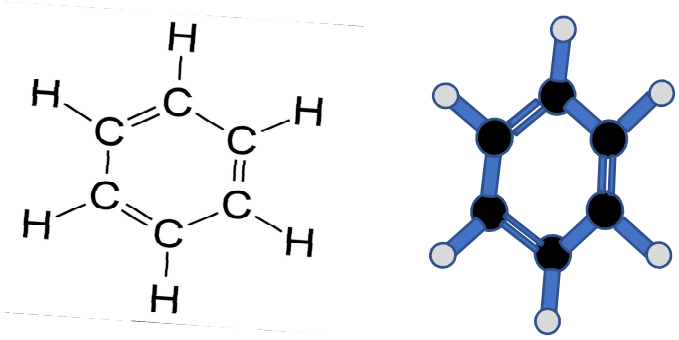

Figure 3. The benzene molecule, an example of a conjugated π-system.

Like the conjugated π-systems, an engineering system can be represented as a network or graph of *n* interconnected elements that resemble a molecule graph of *n* interacting atoms, as described above. The engineering system's components mirror the atoms in the conjugated π-system, and the system components' interrelations resemble

the bonds among atoms in the molecule. This similarity between the structures of engineering systems and molecular graphs inspired a structural complexity measure [10]. This similarity assumes that the engineering system components share the same complexity level — that they are homogeneous, as the atoms in the molecular system.

The calculation of the structural complexity measure's first term, C_1 (individual components' complexities), can be found by a diversity of methods depending on the amount, quality, and availability of the components' historical data. For instance, when information is scarce due to a lack of project history or inadequate documentation in previous projects, probabilistic estimation models based on SMEs' input can be used. If partial information is available, it is possible to obtain a preliminary design by applying Bayes techniques to the blend of data and SMEs' knowledge [43]. With this technique, the database information is first used to construct an *a priori* distribution. As more complementary information about the conjectures is obtained, the initial probabilities are updated using Bayes probabilities, which allows the construction of *a posteriori* distribution. Let $A_1, A_2, \ldots, A_n$, the Bayes rule for any event A and B, such that $P(A_i)>0$, and $P(B)>0$ is defined as $P(A_i|B) = P(A_i)P(B|A_i)/\sum_{i=1}^{n} P(A_i)P(B|A_i)$, where $P(A_i)$ is the probability of an event A_i for all i, representing the possible conjectures about a topic (prior probability), $P(A_i|B)$ is the conditional probability for each event given event B has occurred (posterior probability), and $P(B|A_i)$ are the conditional probabilities for a particular situation. For example, consider a system consisting of $i=5$ components. Let A_i denote the event of assigning a certain complexity value to a component i. Initially, since there is no information about the components, it is usually assumed that these components are statistically independent; this is, the factors causing the complexity of

one component are different from the factors causing the complexity of another component. For simplicity, suppose the components have an equal probability of having the same complexity, $P(A_1),\ldots,P(A_5)$. This probability distribution is known as the uniform distribution, a particular case of the beta distribution with a mean value (or expected value) $E=\alpha/(\alpha+\beta) = 1/(1+1) = 1/2$, where E represents the long-run frequency[3], $\alpha=\beta=1$ are the parameters of the distribution, resulting in $P(A_i),\ldots,P(A_5) = 1/2$. When these probabilities are subjective[4], an SME does not have certainty regarding the long-run frequency associated with the probability E. However, suppose the SME knows the complexity of the first component. In that case, he/she has additional information that allows him/her to update his /her personal probability by changing the beta distribution parameters to $\alpha= 2$ *and* $\beta =1$. Then, $E= \alpha/(\alpha+\beta)= 2/(2+1) = 2/3$, or $P(A_2|A_1) = 2/3$. Once the SME knows the probability of the first and second components, he/she updates the beta distribution parameters to $\alpha= 3$ *and* $\beta =1$. Then, $E= \alpha/(\alpha+\beta) = 3/(3+1) = 3/4$, *or* $P(A_3|A_1A_2) = 3/4$, and so forth. The process can be generalized for a system of n components to obtain each probability as $P(A_n|A_1,\ldots,A_{n-1}) = n/(n+1)$.

A more elaborate way of calculating the individual complexities can be used when previous components' information is available. In this case, the individual components' complexity (α_i) is found as the sum of weighted rank-based individual components: $\alpha_i = \sum_{j=1}^{m} w_j^{(i)} r_j^{(i)}/m$, where $w_j^{(i)}$ is the weight associated with the j^{th}

[3] The long-run frequency is the proportion of the number of times an event occurs in a large number of trials over the number of trials in the long run [73].

[4] Subjective probabilities, also known as personal probabilities, are those probabilities based on people's intuition and experience [73].

factor for the i^{th} component, $r_j^{(i)}$ is the percent rank with respect to the variable $x_j^{(i)}$ in the historical database of similar components [51][52], and m is the size of the component characteristic vector $X=\{x_1, x_2, \ldots, x_m\}$. Note that each weight $w_j^{(i)}$ is found as a function of the development cost/effort of the vector X containing the component's technical characteristics. Eight technical characteristics are suggested: x_1: Measure of performance tolerance, x_2: Measure of performance level, x_3: Component size indicator, x_4: Number of coupled disciplines involved, x_5: Number of variables and physical processes involved, x_6: Component reliability measure, x_7: Existing knowledge of operating principle, and x_8: the extent of reuse/heritage indicator.

The second term C_2C_3 of the complexity measure C in [10] is the interface complexity term, which accounts for a) the interface activity (C_2) and b) the system's topology (C_3). On the one hand, the interface activity is calculated as the sum of the interface complexity between each pair of components i and j ($\beta_{i,j}$) multiplied by the adjacency matrix ($A_{i,j}$): $C_2 = \sum_{i=1}^{n} \sum_{i=1}^{n} \beta_{i,j} A_{i,j}$. The interface complexity between two components is, in turn, calculated as a function of the individual components' complexity and a parameter characteristic of the interface type: $\beta_{i,j}^{(k)} = g\left(c^{(k)}, \alpha_i, \alpha_j\right)$, where $c^{(k)}$ is a constant based on the interface type, and α_i and α_j are the individual complexities of components i and j, respectively. On the other hand, the topological complexity C_3 is the graph's energy normalized by the structure's number of components: $\gamma E(A)$, where $\gamma=1/n$, n is the total number of components, and $E(A)$ is the graph's energy calculated as the sum of the adjacency matrix's singular values [32].

Theoretical and empirical validations of the structural complexity measure in [10]

are provided in more detail in reference [53]. Firstly, the mathematical validation is performed following Weyuker's criteria [54]. Weyuker's criteria are a formal set of computer program characteristics such as implementation, understanding, testing, and maintenance difficulty that enable the objective comparison of SW complexity. Weyuker's criteria were intended originally for SW program complexity measures, and their transferability to systems engineering measures is questionable. It has been found that the criteria do not work well for non-traditional programming languages, and flawed measures have been proved to satisfy the criteria (Type-II error) [71].

Secondly, the empirical validation is not found by directly measuring system complexity. Instead, a hypothesis is formulated to justify an indirect measurement, where the structural system complexity correlates with the system's development effort. The empirical validation was based on chemistry kits consisting of balls and sticks that resemble atoms and bonds in organic molecules. The structural complexity of several molecules represented by the chemistry kits was quantified using the structural complexity measure. Then, subjects were tasked with assembling different molecules representing various complexity levels. The time spent by the volunteers on assembling the molecules (assembly time) represented a measure of development effort. Then, the assembly time was correlated to the system structural complexity by mathematical regression using the functional form: $E = \lambda C^{\xi}$, where E is the estimated design cost/effort, C is the structural complexity measure as described above, and λ and ξ are parameters found by the regression analysis. The first shortcoming of the empirical validation approach described above is that it poses a transferability issue because the development effort is highly dependent on many factors such as historical data, organization processes

and culture, and unique project characteristics. The second shortcoming of the empirical validation is that it did not use a real-world complex engineering system because of the lack of publicly available data.

Another disadvantage of applying the structural complexity measure proposed by [10] lies in calculating the system's individual components' complexity. Even with moderated complex systems, the mathematical procedure previously described becomes intricate, and more straightforward methodologies are preferred. In the examples provided in [10], the individual complexities (α_i) were assumed or estimated using the Technology Readiness Level approach [55]. However, despite the Technology Readiness Level approach's relative simplicity, its application presents implementation challenges such as imprecision of the scale, assessment subjectivity, and components integration, dependencies, and connectivity [56]. The structural complexity measure in [10] is also restricted, quantifying only structural complexity, leaving out functionality, an essential ingredient of system complexity strongly associated with system states and behavior, as stated previously.

2.12 A Spectral Complexity Measure

A set of complexity measures based on spectral analysis and graph theory is presented in [11]. Graphs and networks are interchangeable terms to describe how the interrelationships are occurring in a system [32]. However, strictly speaking, the term network is often used to describe the physical aspect of these interrelationships, while the term graph describes the mathematical abstraction of it. Networks are defined in terms of nodes (system's components) and links (interrelationships). In graph theory, the graph's vertices or nodes represent the system components, and the edges or links are the

interconnections (see Figure 2). The interconnections are usually characterized by probability distributions [32]. Graphs can be described mathematically as a rectangular array of numbers (or matrices), as presented in section 2.1.2.

2.12.1 The Laplacian and normalized Laplacian spectral complexity measures

The Laplacian and normalized Laplacian matrices represent the complex system's graph configuration in addition to the adjacency matrix. The matrix's Laplacian (L) is by definition the difference between the degree matrix (D), which contains the number of edges of each vertex, and the adjacency matrix (A): L=D-A. The normalized Laplacian (L_n) is defined in terms of the Laplacian as $L_n = D^{-½}LD^{-½}$. In general, a graph's energy is the sum of the absolute values of its adjacency matrix's eigenvalues [69]. The set of eigenvalues is called the graph's spectrum. The Laplacian energy is the sum of the absolute value of the difference between the Laplacian spectrum elements and twice the ratio between the number of edges and vertices. The Laplacian spectrum is characterized as having non-negative elements. Consequently, their sum is non-zero. Due to this characteristic, the Laplacian energy is equivalent to the graph's energy for regular graphs but not for non-regular graphs (singular) [70]. Then, in general, the graph's Laplacian energy is greater or equal to the graph's energy. As seen in section 2.7, the graph's energy concept has its origins in molecular chemistry.

2.12.2 Natural Connectivity and Laplacian Natural Connectivity Spectral Complexity Measures

The other two network measures used in [11], natural connectivity and Laplacian natural connectivity, quantify network robustness. Network robustness here is understood

as the network's ability to endure failures. Robustness is a desired characteristic in critical infrastructure networks such as communication, power distribution, and railroad systems. The first measure, the natural connectivity, is defined as the logarithm of the average sum of closed walks of all lengths [72]. A network's walk is an alternating sequence of vertices and edges, in which length is the number of edges that it passes. A closed walk is a walk that begins and ends at the same vertex. In the network provided in Figure 4, *a1b2c2b3c5d4ba* is a length 7 close walk.

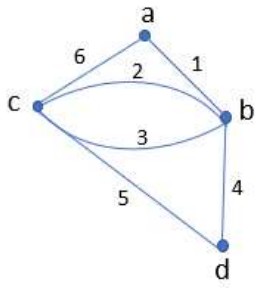

Figure 4. Example of a network walk

The second measure, the Laplacian natural connectivity, uses the eigenvalues of the weighted normalized Laplacian matrix. The generalized spectral complexity measure proposed in [11] considers the concepts reviewed above and simultaneously quantifies the Laplacian graph energy, natural connectivity, and Laplacian natural connectivity. Unlike the structural complexity measure proposed in [10], the generalized spectral complexity measure can be applied to complex systems with either homogeneous or heterogeneous graphs. The spectral complexity measure was evaluated in the study of the Tactical Aircraft System of Systems [57]. The first apparent shortcoming of this spectral complexity measure is that it evaluates only the structural complexity of static complex systems. In general, the spectral measures focused heavily on the system configuration

and its robustness, which indicate system complexity. However, spectral measures miss the complexity of the individual components. When applied to a complex system, the individual spectral measures' trends may differ, depending on the system's characteristics, such as distribution, number of links associated with a node, network density, and interconnectivity levels. It may be difficult for some systems to establish a common trend because some of the individual spectral measures may increase while others decrease. Under these circumstances, inferring a net trend or net network behavior is not always an easy proposition. Besides, the individual spectral measures' opposite trends may counteract each other, leading to a small complexity value, which may not correspond to what is, in reality, a highly complex system, or vice versa.

2.13 Summary

In this chapter, fundamental concepts about complex systems and several complexity measures have been reviewed. Many measurements available have been used to quantify three aspects of engineering design: the design problem, the design product, and the design process—however, most of these complexity measures apply to particular processes of the product life-cycle's and are related exclusively to either structural or functional complexity. Although system behavior is a fundamental concept to define complex systems, it has been omitted in current complexity measures. This research considers the omission and designs a complexity measure integrated with the MBSE approach to quantify system structure, function, states, and behavior.

Chapter 3:Research Methodology

To answer the Research Questions and achieve the Research Objectives considered in the previous chapter, a mathematical model was formulated that combined graphs theory, signal processing, physics principles, and stochastic models. In building the mathematical model, the goal was to develop a scientific understanding of system complexity and explore the quantification of system behavior, structure, function, and states. The mathematical model of the complexity measure was developed using a methodology consisting of the following stages [76]:

1 Articulate the problem abstraction. In this stage, the understanding of the system's essential attributes is translated into mathematical equations. This stage is comprised of:
    a. Reviewing the scientific principles and theory to be used in the model (Section 3.1.1).
    b. Determining which attributes of the system must be assessed and their relationships (Section 3.1.2).
    c. Defining the mathematical equations related to the attributes of interest (Section 3.1.2). The mathematical derivation to formulate the proposed complexity measure implements:
        i. Analogies from molecular physics, graph theory, and signal processing.
        ii. Stochastic models to account for system uncertainty.
2 Solve the mathematical problem. This stage consists of formulating a testable solution that likely solves the stated problem (Section 3.1.3). For that purpose,

a computational model-based system architecture is developed to generate the validation data under controlled conditions. An MBSE approach was selected in this stage, closely following the OOSEM methodology described in Section 2.5.

3 Interpret the solution. In this stage, the mathematical model's efficacy in solving the system complexity problem (in terms of behavior, structure, functions, and states) is analyzed and compared against traditional solutions.

3.1 Development of Complexity Metric

The first research objective, "Develop a complexity measure based on a system structure, function, and behavior approach," was achieved using a deductive mathematical development method. Section 3.1.1 provides the theoretical framework based on quantum mechanics postulates and signal processing. The proposed complexity measure's derivation is presented in Section 3.1.2.

The second research objective, "Integrate the novel comprehensive complexity measure with a model of the system," was achieved by developing a test system model based on an MBSE, closely following the OOSEM method explained in Section 2.5. The model development is presented in Section 3.1.3. Section 3.1.4 covers the application of the proposed complexity measure to the system's model.

The third research objective, "Determine if a comprehensive complexity measure approach provides advantages over traditional approaches in measuring complexity," was achieved by analyzing the integration of the mathematical formulation and the system model, presented in Section 3.1.4.

3.1.1 Theoretical Framework

3.1.1.1 Overview of Molecular Orbital Theory and Networks

The proposed complexity measure was developed based on the first three postulates of quantum mechanics [31]:

Postulate 1. "The state of a system is completely specified by the wave function or state function ψ" [31].

According to the first postulate, if a system exists in the space of physical states (also known as Hilbert space [40]), there is a function that fully defines its existential condition in that space at a given moment in time. It also implies that the physical state of a system at each time forms a state vector or state function. In the Hilbert space, any linear combination of states is again another state (superposition principle). Moreover, a state can be represented as a linear combination of basis vectors or basis functions in the Hilbert space.

Postulate 2. "To every observable in classical mechanics, there corresponds a linear, Hermitian operator in quantum mechanics" [38].

According to the second postulate, each of the system's measured physical properties or attributes correspond to a mathematical operator (Hermitian operator) that operates in the Hilbert space on the wave function. Furthermore, if a particular physical property of the system is measured at each point in time, the collection of obtained measurements can be described by a probability distribution function [37]. Statistical predictions of system behavior are possible if this probability distribution is known.

Postulate 3. "If a state function ψ describes a system, the only possible results of measuring the observable associated with operator $\hat{A}$ are the eigenvalues of the operator

Â that satisfy the eigenequation:

$$\hat{A}\psi = a\psi \tag{9}$$

where $\hat{A}$ is the operator, ψ is the state function, and a are the eigenvalues of the operator $\hat{A}$. According to the third postulate, because the observables or measures of a physical attribute are real numbers, the operators' eigenvalues representing the physical attribute must be real numbers. The probability $P(a_k)$ of finding a particular eigenvalue a_k is given by the square modulus of the scalar product of the normalized state $|\psi\rangle$ by the corresponding normalized eigenvector: $P(a_k) = |\langle a_k|\psi\rangle|^2$. Furthermore, operators with real eigenvalues are Hermitian [40]. When an operator operates on a wave function, it generally results in a different wave function. However, when the resulting waveform is a scaled version of the operated waveform, the new waveform is called an eigenstate. In that case, the measurement of the observable is the eigenvalue for that eigenstate. A Hermitian operator's eigenstates span the Hilbert space, forming a basis [38], [40]. Orthogonality is not a necessary condition for the eigenstates of a Hermitian operator.

3.1.2 System Complexity Measure Incorporating System Behavior

3.1.2.1 From Quantum Mechanics to System Behavior

The Schrödinger equation discussed in Section 2.7 applies to all physical systems as long as the system is well-defined with a known Hamiltonian, making it one of the most general physics equations. However, the Schrödinger equation rarely has an exact solution. Even at a molecular level, it is virtually impossible to predict the exact behavior of a simple system consisting of one or more particles interacting with each other when a perturbation is introduced to the system. Solutions, such as the HOM proposed by Hückle

(Section 2.7), require some approximations.

Natural phenomena such as electrical current and moving objects have been modeled exhaustively by engineers and scientists as continuous-time systems using differential equations [62]. The dual of differential equations for discrete-time models is the difference equations. The objective of solving the differential equations or difference equations that describe a system is to find the system's transfer function. The system's transfer function (also known as system function) relates the system's output to its input [41]. If the system's function is defined, it is possible to predict the system's behavior for a particular state when a known input is applied to the system. A transfer function with many variables is usually an indication of system complexity. Differential or difference equations work well for relatively simple systems. For complex systems or non-linear systems, analytical solutions are no longer viable, and approximations or algorithmic computation are likely necessary.

However, according to quantum mechanics' general formalism, basis functions are not confined to describing parametric or differential equations and can be chosen to describe behavioral states. As stated in Section 3.1.1.1, a system's behavior is described by a linear combination of possible states or basis functions ψ_i (see Eq. (3)). The Hilbert space properties, such as scaling and addition, allow the mathematical manipulation of the basis functions ψ_i to model the system's attributes of interest. If ψ_i represents a particular system state, the total system state ψ can be expressed as a linear combination of states ψ_i scaled by P_i, a factor representing the probability that the system is in that state:

$$\psi = \sum_{i=1}^{n} P_i \psi_i = P_1\psi_1 + P_2\psi_2 + \cdots + P_n\psi_n \quad (10)$$

where ψ *is* the system's total behavior, ψ_i is a normalized orthogonal basis function representing each of the n possible system states, P_i is the cumulative probability distribution of finding the system in a particular state during a given period, and i is an index varying from 1 to n.

The probability distribution or Probability Density Function (PDF) of a random variable X is a function $f(x)$ defined as the probability that X takes a value in an interval limited by the parameters x_1 and x_2 [43]:

$$P(x_1 \leq X \leq x_2) = \int_{x_1}^{x_2} f(x)dx \quad (11)$$

where $f(x)$ is the PDF, X is a random variable, and x_1 and x_2 are any two numbers that limit the interval of possible values of X. The PDF has two particular propositions if X is a continuous random variable. The first proposition establishes that the probability of an interval does not depend on whether either of the limits of the interval are included:

$$P(x_1 \leq X \leq x_2) = P(x_1 < X \leq x_2) = P(x_1 \leq X < x_2) \quad (12)$$

The second proposition establishes that the probability of $X=x_1$ is zero:

$$P(X = x_1) = P(x_1 \leq X \leq x_1) = \int_{x_1}^{x_1} f(x)dx = 0 \quad (13)$$

or in other words, the probability assigned to any particular value is zero.

If X is a discrete random variable, the probability distribution is known as a Probability Mass Function (or a Probability Density Function (PDF) if X is a continuous

random variable), defined as the probability that the random variable X takes the value x within a set S:

$$P(X = x) = P(\forall s \in S: X(s) = x) \tag{14}$$

The PDF is related to the Cumulative Distribution Function (CDF). The CDF of a random variable X is defined as [36]:

$$F(X) = P(X \leq x) = \begin{cases} \int_{-\infty}^{x} f_X(y)dy, & if\ X\ is\ continuous \\ \sum_{k \leq x} p_X(k), & if\ X\ is\ discrete \end{cases} \tag{15}$$

where $F(X)$ is the CDF, X is a random variable, and x is a specified number such that $y \leq x$. The following expression relates the PDF and CDF:

$$P(x_1 \leq X \leq x_2) = F(x_2) - F(x_1) \tag{16}$$

To illustrate the PDF and CDF concepts, suppose that the complexity of implementing a function in software is described by a continuous random variable C. Also, suppose that the available technical information and performance requirements regarding the function are sparse. Additionally, suppose that the SMEs estimate the complexity X of implementing the function would take some value x in the interval [0 100]. Finally, considering the initial information sparseness, the SMEs indicate that any values of X in that interval are equally probable. In other words, the distribution of X has a constant PDF (uniform distribution). Then, $f_X(x)$ can be expressed as $f_X(x)=c$. The constant c can be calculated using (11):

$$P(x_1 = 0 \le C \le x_2 = 100) = \int_{x_1}^{x_2} f(x)dx \int_0^{100} c\, dx = 100c = 1 \qquad (17)$$

consequently, c=1/100=0.01. Then, the PDF of the random variable X can be rewritten as $f_X(x)$= 0.01, if $0 \le x \le 100$. Finally, the CDF can be determined from (15) as:

$$F_X(x) = P(X \le x) = \int_{-\infty}^{x} f_X(y)dy$$
$$= \begin{cases} 0, & if\ x < 0 \\ \int_0^{100} 0.01\, dy = 0.01x, & if\ 0 \le x < 100 \\ 1, & if\ x \ge 100 \end{cases} \qquad (18)$$

3.1.2.2 Formulation of the Proposed System Complexity Measure

The proposed complexity measure to quantify system complexity is defined as:

$$C = \sum_{i=1}^{n} \mu_i |P_i \psi_i|^2 \qquad (19)$$

where C quantifies the complexity of the system's primary function or operation, μ_i is a factor that represents the complexity of the methods associated with state i, P_i is the probability distribution of finding the system in a particular state i at a given time, and i is an index varying from 1 to n possible states. C is a unitless scalar quantity.

When the system performs several primary functions or operations, the system's complexity is calculated for each main operation. Then, the total system complexity is the sum of the individual system complexities evaluated for each operation. In that case, the total system complexity is calculated as:

$$C_{Sys} = \sum_{k=1}^{m} C_k = \sum_{k=1}^{m} \{C_1, C_{2,\ldots,} C_m\} \tag{20}$$

where C_{Sys} is the total complexity of the system, and C_k=C_1, C_2, ..., C_m are the system complexities evaluated at each main system operation, m is the maximum number of primary operations, and k is an index varying from 1 to m. The larger the number of system operations and functions, the larger the total system complexity.

To calculate μ_i in (19), let M_1, M_2, ..., M_k be the methods or functions of the i^{th} component (block or class) of a state, with associated complexities π_1, π_2, ..., π_k. The calculation of the individual functional complexities λ_i of the i^{th} component is the total of the complexities of all the different functions or methods implemented within that component, block, or class:

$$\lambda_l = \pi_1 + \pi_2 + \cdots + \pi_k = \sum_{j=1}^{k} \pi_j \leq k \tag{21}$$

where λ_l is the individual functional complexity of the i^{th} component, and π_j is the complexity of the method M_j, defined as a function of the probability that M_j successfully meets a requirement [8]:

$$\pi_j = -log_2(\rho_r) \tag{22}$$

where π_j is the complexity of the method M_j and ρ_r is the probability of successfully satisfying a requirement. The probability ρ_r is given by:

$$\rho_r = prob[a \leq r \leq b] = \int_a^b f(r)dr \tag{23}$$

where ρ_r is the probability successfully satisfying a requirement, r is the predicted response, $f(r)$ is the probability density function of r, and a, b are the requirement's tolerance limits.

Notice that λ_l can also be defined as a probability of failing a requirement, such that $\pi_j = (1-q_j)$. Then, (21) can be reformulated as:

$$\lambda_l = (1 - q_1) + (1 - q_2) + \cdots + (1 - q_k) = \sum_{j=1}^{k} (1 - q_j) \leq k \tag{24}$$

where λ_l is the individual functional complexity of the i^{th} component, and q_j are the individual probabilities of unfulfilling a requirement.

The total functional complexity μ_i associated with a state i is defined as the sum of all the individual functional complexities of all the components intervening in an operation:

$$\mu_i = \sum_{l=1}^{n} \lambda_l \tag{25}$$

where μ_i is the total functional complexity associated with state i, and λ_l is the ith component's individual functional complexity.

In the particular case when each function or method satisfies a requirement, $\pi_j = 1$, then:

$$\lambda_l = k \tag{26}$$

where λ_l is the individual functional complexity of the i^{th} component (block or class), and k is the total number of methods of the i^{th} component associated with a state. Then (25) can be simplified as:

$$\mu_i = nk \tag{27}$$

where μ_i is the total functional complexity associated with a state, k is the total number of methods of the i^{th} component associated with a state, and n is the total number of elements.

When the probabilities P_i are given in terms of event probabilities p_i (point estimate) instead of probability distributions ($P_i = p_i$), the complexity measure proposed in (19) can be redefined as:

$$C = \sum_{i=1}^{n} \mu_i |p_i \psi_i|^2 \tag{28}$$

where C is the complexity of the system evaluated at the primary system's function or operation, μ_i is a factor representing the complexity of the methods or functions associated with a state calculated with (25), p_i is the probability of finding the system in a particular state at a given time, and i is an index varying from *1* to n possible states.

3.1.3 Developing a Model for a Test System using an MBSE Approach

An Automatic Teller Machine (ATM) is selected as the system to model. The MBSE diagrams of Figure 5 to Figure 9 describe the ATM system model. The model

diagrams used to describe the ATM system contain enough detail about the system to illustrate how the proposed complexity measure is applied to the system model but do not contain excessive details, to avoid clutter that might obscure the methodology. In terms of requirements, the system must have three principal system functions or operations: 1) Withdrawal, 2) Transfer, and 3) Deposit Transactions. A typical ATM transaction session consists of reading a debit card and authenticating the customer. The authentication consists of requesting the pin from the customer and verifying it. After the authentication, the customer selects the type of transaction, the transaction amount, and the type of account(s) involved in the transactions. Then the system executes the transaction after verifying the balance and updating the accounts involved in the transaction.

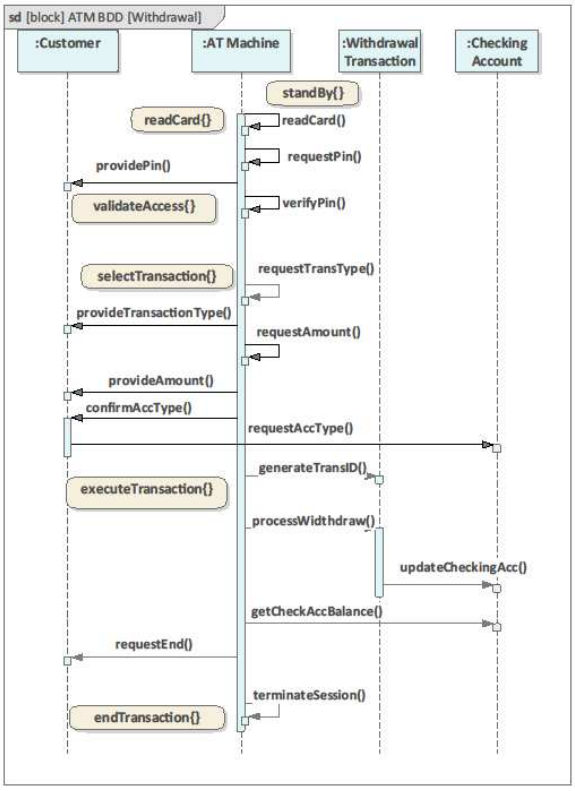

Figure 5. ATM SD of Withdrawal from the Checking Account (©2021 IEEE).

The events occurring in the Withdrawal Transaction from the Checking Account

are described in the Sequence Diagram (SD), as illustrated in Figure 5.

From the SD, four main system elements (blocks or classes) can be identified in the Withdrawal Transaction operation:

1) Customer

2) ATM

3) Withdrawal Transaction

4) Checking Account

In the SD, the elements' lifelines and the communication flow (messages) between elements are represented with dotted lines and solid arrow lines between lifelines, respectively. The messages appear as a sequence of events.

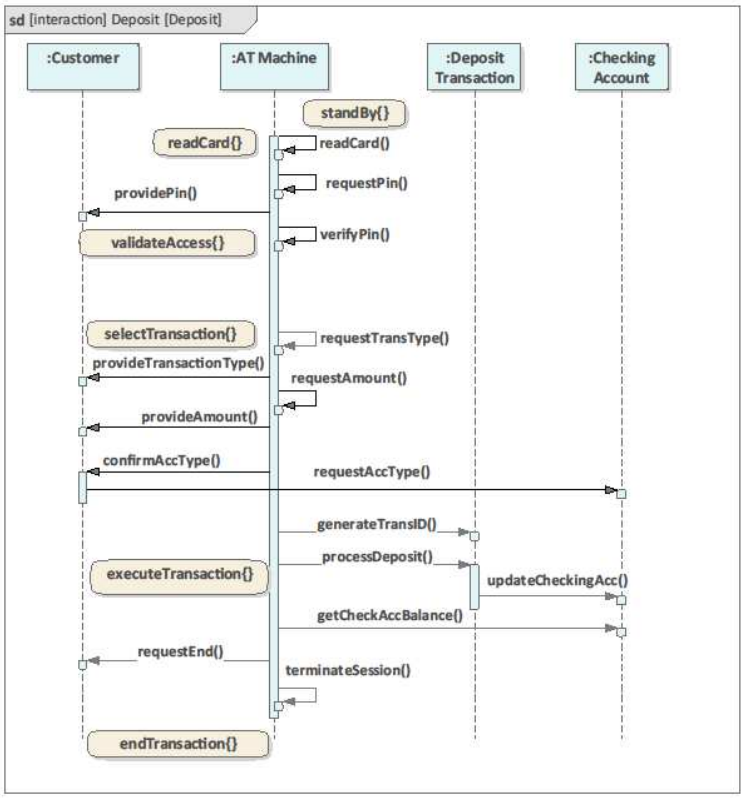

Figure 6. SD – Deposit to the Checking Account (©2021 IEEE).

Additional SDs for the Deposit and Transfer operations can be constructed following the same process used for constructing the Withdrawal SD. The resulting SDs for the Deposit and Transfer Transactions are shown in Figure 6 and Figure 7.

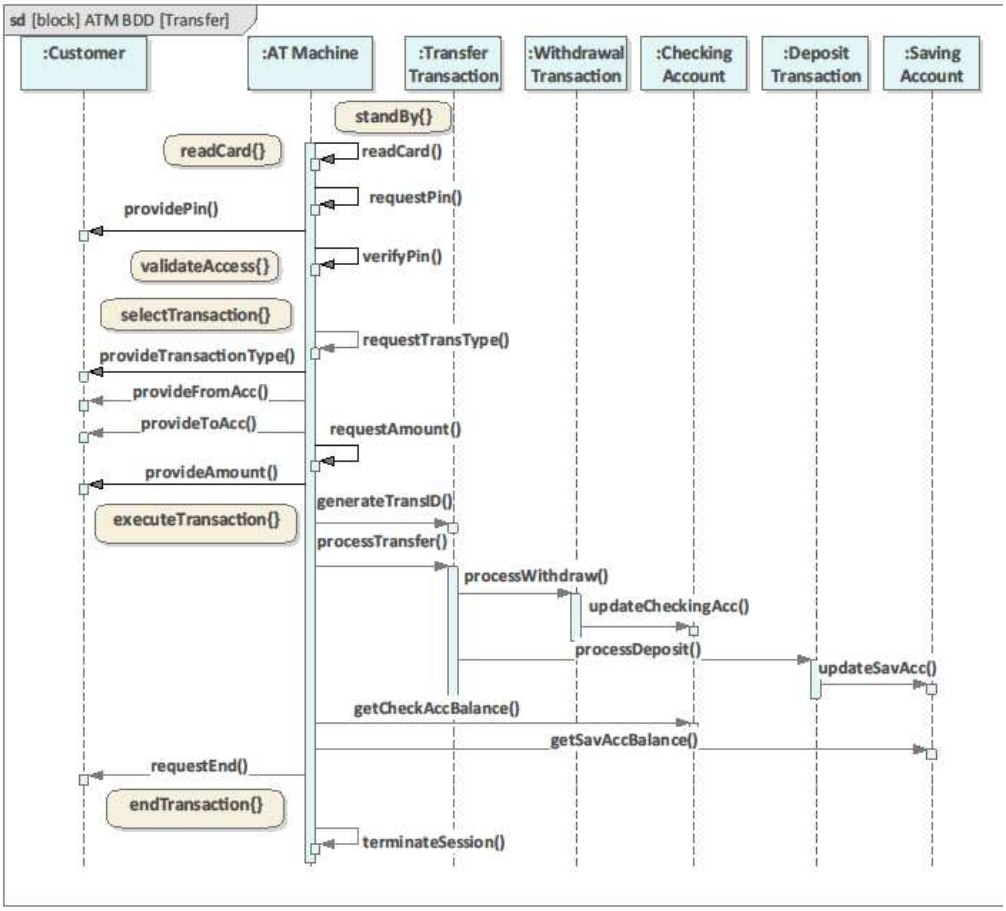

Figure 7. SD -Transfer from Checking Account to Saving Account (©2021 IEEE).

The next step is to allocate the system elements (blocks) and determine their relationships in the Block Definition Diagram (BDD), as illustrated in Figure 8.

The relationships between elements are represented as connections in the BDD. These types of connections are known as associations. Blocks' behaviors may use these

associations to request operations or obtain values from the other blocks. The associations between blocks may have a direction to indicate how the information flows and a multiplicity number to indicate how many associations are connecting the associated blocks.

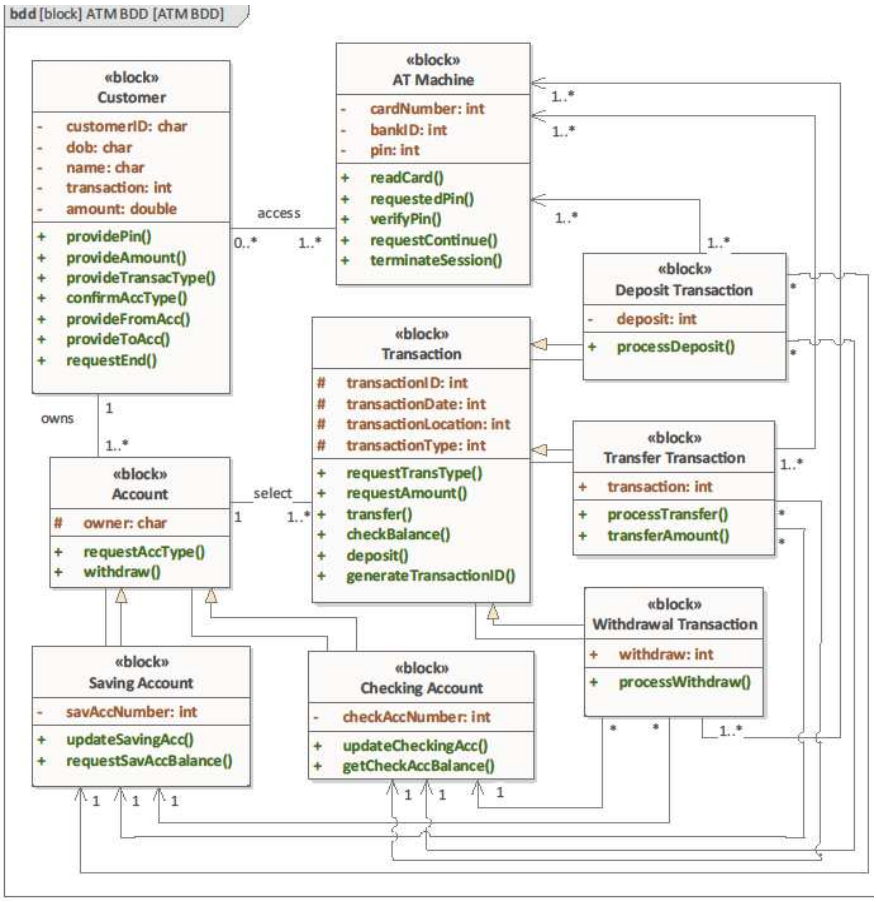

Figure 8. ATM System BDD (©2021 IEEE).

The events or actions sequence and direction depicted in the Transfer from Checking Account to Saving Account SD (Figure 7) define how the interactions between blocks occur in the BDD (Figure 8). The requests of operations or data between blocks trigger actions or functions to be performed by the blocks. These functions initially described in the SDs are now shown in the lower boxes of the BDD blocks (Figure 8).

As part of the modeling process, the BDD assembly in Figure 8 may help to discover new events and elements that the architect did not foresee at the beginning of the modeling process. These elements and events may be necessary to implement the intended operations and satisfy the requirements. For example, suppose that initially, the architect did not include a Saving Account in the model and realized that to enable a Transfer Transaction, both a Checking and a Saving Account are necessary. Different accounts may serve different purposes for customers, giving them the flexibility needed to perform diverse operations in the ATM system. Then, the architect must include the block Saving Account in the BDD of Figure 8. Furthermore, the architect has to review the SD of Figure 7 and include a lifeline for the Saving Account to synchronize the model information of both the BDD and SD (Figure 8 and Figure 7). In general, discoveries and dependencies affecting a model diagram have an impact on other diagrams. As more model views are created, all of the corresponding diagrams necessary to describe the system must be synchronized. Modeling the system is then an iterative process that constantly refines the architect's understanding of the system. Not surprisingly, modeling decisions adopted in one diagram have a ripple effect in the remaining system diagrams. The MBSE toolset and OOSEM method help keep the system model's consistency while this fine-tuning occurs.

Six primordial system states can be identified from the system SDs (Figure 5, Figure 6, and Figure 7):

1) standby

2) read card

3) validate access

4) select transaction

5) execute transaction

6) end transaction

These six states are depicted in the State Machine Diagram (Figure 9).

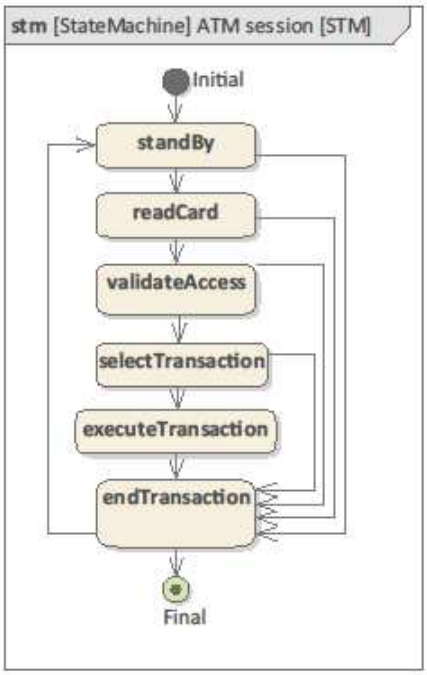

Figure 9. ATM State Machine Diagram (©2021 IEEE).

The State Machine Diagram (SMD) shows the behavior of the system as it transitions between states. The arrows in the SMD indicate a transition's direction from the initial state to the final state.

The set of primordial states in Figure 9 constitute the space state, defined mathematically as:

$$S = \{\psi_1, \psi_2, \psi_3, \psi_4, \psi_5, \psi_6\} \quad (29)$$

where S is the space state, ψ_1 is the standby state, ψ_2 is the read card state, ψ_3 is the validate access state, ψ_4 is the select transaction state, ψ_5 is the execute transaction state, and ψ_6 is the end transaction state.

Applying the proposed complexity measure to the ATM system requires a more detailed definition of the state functions in the system's space state. In general, determining the system's states is a highly customized activity that depends on attentive observation of how the SOI behaves while operating. In the ATM system case, any transaction operation (Withdraw, Deposit, or Transfer) starts when a customer arrives at the ATM and initiates a transaction. The activity finishes once the transaction is accomplished. Then, a new customer arrives, and a new activity starts again. The customer arrival activity is dynamic, as it depends on time. The duration of the customer arrival activity is also random because of the uncertainty involved in predicting future arrival events. Accounting for this uncertainty in the process requires defining the activity duration in terms of a random variable with an associated probability distribution (see (11)). Assuming that successive customer transactions are independent and identically distributed random variables, the customer arrival during transaction processes can be described by a Poisson distribution. Poisson processes describe a series of random discrete events occurring with a known average time between events. In the ATM system, a customer transaction does not depend on previous customer transactions. The assumption of a Poisson distribution here to describe the ATM process is appropriate and mathematically convenient [77]. There is sufficient evidence that the Poison distribution

accurately describes the distribution of the number of arrivals per period [22]. The Poisson probability mass function is defined as follows [74]:

$$p_X(x;\lambda) = \frac{e^{-\lambda}\lambda^x}{x!}, \qquad x = 0, 1, 2, \ldots \tag{30}$$

where the $p_X(x; \lambda)$ is the probability mass function of the random variable X, λ is a positive number representing the average number of events in a given interval or time unit, e is Euler number (the base of the natural logarithm), x denotes the values of the integer-valued random variable, and $x!$ is the factorial of x. Because $\lambda>0$, $p_X(x; \lambda)>0$ for all possible x values. The probability of finding the ATM system in a particular state for a period (see (29) and (11)) can be expressed in terms of the Poisson CDF, defined as [75]:

$$F(x;\lambda) = p_X(X \leq x) = \sum_{i=0}^{x} \frac{e^{-\lambda}\lambda^i}{i!} \tag{31}$$

At this point, Research Objectives 1 and 2 have been accomplished. Research Objective 3, determining the advantages of using the proposed complexity measure and associated modeling approach, is achieved by analyzing the results of integrating the proposed complexity measure and the system model. This analysis is explained in the following section.

3.1.4 Advantages of The Proposed Complexity Measure over Traditional Complexity Measures.

The complexity calculation defined in (11)-(20) allows the identification of significant complexity elements in the system because it quantifies the number of

methods (or functions) scaled by their associated probabilities. The advantage of applying the proposed complexity measure is that it measures functional complexity while simultaneously accounting for the model's structural complexity, which already includes the three structural elements: components, interrelationships, and topology. When the model's structure is altered, its associated functions are impacted, which is immediately sensed by the complexity measure. The proposed complexity measure also accounts for system state while accounting for individual functional complexities (the term μ in (19)). As reviewed in Chapter 2, traditional complexity measures do not simultaneously account for structural and functional complexity and do not account for system behavior.

Identifying highly complex elements by the proposed complexity measure early in the design process is advantageous, as justified in Section 2.2. The complex components constitute the bulk of the design and development efforts and underestimating them may lead to inaccurate project estimates. Popular estimation techniques and management practices, such as the Critical Path Method, Program Evaluation and Review Technique, and Work Breakdown Structure [42], cannot deliver accurate estimates because of factors such as configuration uncertainty, insufficient historical data, incomplete requirements, prototyping inability, system size misjudgment, and especially complexity oversight.

In the SW domain, the number of Lines of Code (LOC) has been used to quantify complexity. This measurement is likely to mislead the development effort because writing code is highly subjective. The LOC to implement a function depend on the programming language, programmer style, programmer experience, and programmer approach to the problem. Complexity measures like the LOC evaluate only structure (i.e., just size), ignoring the function's complexity. In the proposed complexity measure, an

element's individual functional complexity is defined in terms of the probability of successfully fulfilling a requirement, which is related to information content, as explained in Section 2.6.

Also, as reviewed in Chapter 2, some complexity measures are challenging to implement in practical systems. The simplifications of the proposed complexity measure in (19) and (20) are intended to save time and calculations when systems are vast. Additionally, when there is a lack of historical information or is impossible to realize system prototypes, the probability distribution in (11) cannot be readily assumed. However, the formulation of the proposed complexity measure and the iterative nature of the modeling process allows for the assumption of an initial probability distribution and its redefinition once the model matures. For example, the architect can initially assume a simple probability distribution such as the uniform probability distribution, where the probability of successfully satisfying a requirement is calculated as a rectangular area limited by the design tolerance parameters [a, b] and the response range [r_1, r_2]. Then, (23) is reduced to:

$$\rho_r = \left(\frac{max(a, r_1) - min(b, r_2)}{r_2 - r_1} \right) \tag{32}$$

where ρ_r is the probability of success of satisfying a requirement, a and b are the parameters of the design tolerance interval, and r_1 and r_2 are the parameters of the response range.

From Chapter 2, several traditional complexity measures do not even consider time. The proposed complexity measure is dynamic because it is essentially based on

system states. By definition, a state fully defines the system's behavior at some point in time and therefore has dynamism. Dynamism is a fundamental characteristic of complex systems because it accounts for the interdependence of components and functions and how they impact system behavior. The proposed complexity measure's holistic approach accounts for the mutual influence of components, interactions, configuration, functions, and states and reflects that in the mathematical formulation and the modeling process. New or modified requirements after an engineering project has started are not rare events. In complex systems, these changes translate into system redesigns with modified structures, functionality, and states. Under new conditions, the system behavior can change significantly. Because system behavior is not necessarily static but may change over time, the proposed complexity measure couples states and interactions stochastically, rather than deterministically. Therefore, the proposed complexity measure is by design sensitive to structural, functional, or behavioral changes in the system, which may arise in both the short and long term.

Chapter 4: Results

In this chapter, two test cases are presented using the ATM system model developed in Chapter 3. In the first test case, the proposed complexity measure is applied to the ATM system's model. The system's complexity is also quantified using one of the top-of-the-line structural complexity measures reviewed in Section 2.11 [10]. In the second test case, new system requirements are introduced to precipitate modifications in the system: the first requirement originates system behavior changes, while the second affects functionality. Once the architectural modifications are performed on the system to meet the new requirements, the ATM system complexity is re-evaluated with the proposed comprehensive complexity measure and the structural complexity measure.

4.1 Test Case I: Baseline

As discussed in Section 3.1.3, the ATM space state S is constituted by the standby state (ψ_1), read card state (ψ_2), validate access state (ψ_3), select transaction state (ψ_4), execute transaction state (ψ_5), and end transaction state (ψ_6) (see (29)). As justified in Section 3.1.3, Poisson processes characterize the ATM states (see (30) and (31)). Suppose customers arrive at the ATM at an arbitrary average rate $\lambda=20$ customers per time unit. Also, suppose that per system performance requirements, the system should resolve out of standby state such that it would allow the arrival of no more than 10 percent of customers. Then, the quantile of customers arriving while the system is in the standby state (ψ_1) is less than $x_1=10$ percent. Then, the probability that less than 10 percent of customers find the system in the standby state (ψ_1) is computed by substituting these values in the Poisson CDF defined in (31):

$$F(10;20) = p_X(X \leq 10) = \sum_{i=0}^{10} \frac{e^{-20}20^i}{i!} = 0.01 \tag{33}$$

Similarly, per system performance requirements, the system should resolve out of read card state such that it would allow the arrival of no more than 20 percent of customers. Then, the quantile of customers arriving while the system is in the read card state (ψ_2) is less than $x_2=20$. Substituting these values in (31) again, the probability that less than 20 percent of customers use the system in read card state (ψ_2) is:

$$F(20;20) = p_X(X \leq 20) = \sum_{i=0}^{20} \frac{e^{-20}20^i}{i!} = 0.56 \tag{34}$$

The exact process described above can be followed to find the probabilities for the remaining system states. The complete set of quantiles of customers arriving for each of the remaining system states is given per performance requirements as:

$$x = \{x_1, x_2, x_3, x_4, x_5, x_6\} = \{10, 20, 15, 30, 20, 10\} \tag{35}$$

Thus, substituting the x values specified in (35) to evaluate the corresponding Poisson CDFs results in the following set of probabilities:

$$\begin{aligned} P &= \{P_1, P_2, P_3, P_4, P_5, P_6\} \\ &= \{F(10{:}\,20), F(20{,}20), F(15{,}20), F(30{,}20), F(20{,}20), F(10{,}20)\} \\ &= \{0.01, 0.56, 0.15, 0.98, 0.44, 0.01\} \end{aligned} \tag{36}$$

Substituting the resulting probability values of (36) in (19), the total system complexity is:

$$C = \sum_{i=1}^{n=6} \mu_{Transaction_type} |P_i \psi_i|^2 = \mu_{Transaction_type} \sum_{i=1}^{n=6} |P_i \psi_i|^2 \quad (37)$$

$$= \mu_{Transaction_type} \{ |0.01 * 1|^2 + |0.56 * 1|^2 + |0.15 * 1|^2$$

$$+ |0.98 * 1|^2 + |0.44 * 1|^2 + |0.01 * 1|^2 \}$$

$$= \mu_{Transaction_type} \{1.49\}$$

Note that the resulting complexity in (37) is a scalar quantity, the magnitude of the linear combination of all the system state functions. In the state function space, the basis functions ψ_i are orthonormal; each has a magnitude of one and is orthogonal to the others, forming a Hilbert space [40]. Orthogonality among basis functions is usually chosen by convenience but is not strictly required. The concept of Euclidian geometric vector space to normed linear function space was extended here. That is to say, rather than a set of geometric vectors, a set of basis functions to represent states was considered. Instead of unit basis vectors, unit basis functions ψ represent arbitrary functions in the state space. Any function in the state space can be represented as a linear combination of scaled versions of the unit basis functions ψ projected on the directions of the state dimensions where the unit basis functions reside.

If each function or method satisfies a requirement, then $\pi_j = 1$ in (22) and (27) can be used to calculate the total functional complexity associated with a state. The number of functions or methods of the i^{th} component, block, or class associated with a state can be determined directly from the BDD (Figure 8). The number of methods or functions of each component during the Withdrawal operation from the Checking Account is summarized in Table 1:

Item	Component	Number of functions
1	Customer	5
2	AT Machine	6
3	Checking Account	3
4	Withdrawal Transaction	2
	Total number of functions $\mu_{Withdrawal}$	16

Table 1. Number of Functions per Component (Withdrawal)

Then, according to (37), the calculation of the system complexity measure under the Withdrawal operation from the Checking Account is:

$$C_{withdrawal} = \mu_{Withdrawal} \sum_{i=1}^{n} |P_i \psi_i|^2 = 16 * 1.49 = 23.8 \quad (38)$$

The calculation of the system complexity for the Deposit and Transfer operations follows the same methodology as above. The total number of methods or functions of each component during the Deposit and Deposit to Checking transfer operations are found by applying (27) to the SDs of Figure 6 and Figure 7, shown in Table 2 and Table 3, $\mu_{Deposit}=16$ and $\mu_{Deposit\ to\ Checking}=20$, respectively.

Item	Component	Number of functions
1	Customer	5
2	AT Machine	6
3	Checking Account	3
4	Deposit Transaction	2
	Total number of functions $\mu_{Deposit}$	16

Table 2. Number of Functions per Component (Deposit)

Item	Component	Number of functions
1	Customer	6
2	AT Machine	6
3	Transfer Transaction	2
4	Withdrawal Transaction	1
5	Checking Account	2
6	Deposit Transaction t	1
7	Saving Account	2
	Total number of functions $\mu_{Transfer}$	20

Table 3. Number of Functions per Component (Transfer)

Then, the system complexity measure for the Deposit operation is found by substituting the value of $\mu_{Deposit}$ in (37):

$$C_{Deposit} = \mu_{Deposit} \sum_{i=1}^{n} |P_i \psi_i|^2 = 16 * 1.49 = 23.8 \quad (39)$$

Notice that the Withdrawal and Deposit operations have the same complexity because they involve the same functionality. Similarly, the system complexity for a Transfer Transaction from Checking Account to Savings Account is:

$$C_{Transfer\ checking\ to\ saving} = \mu_{Transfer} \sum_{i=1}^{n} |P_i \psi_i|^2 = 20 * 1.49 = 29.8 \quad (40)$$

Notice that the Transfer operation is the most complex of the transactions because it involves both the Checking and Saving Accounts. The ATM system represents a general case because it has more than one primary function. Hence, the total system complexity is quantified as the sum of the system's partial complexities when it exercises each primary function. According to (20), when the system performs more than one essential function, the total system complexity is the sum of the complexities observed when the system operates under each function. Then, the total ATM system complexity can be found by adding (38), (39), and (40):

$$C_{Sys} = \sum_{i=1}^{m} \{C_1, C_{2,\dots,} C_m\} = 2 * \{C_{Withdrawal} + C_{Deposit} + C_{Transfer}\} \quad (41)$$

$$= 2 * \{23.8 + 23.8 + 29.8\} = 154.8$$

A factor 2 in (41) indicates that both Checking and Saving Accounts present the

same complexity under each transaction.

Finally, to perform a benchmarking between the proposed complexity measure and a state-of-the-art complexity measure, Sinha and de Weck's structural measure explained in Section 2.11.2 was selected [10]. Thus, the ATM system total structural complexity is calculated using (8):

$$C_{struct} = C_1 + (C_2 * C_3) = 21 + (84 * 1.1) = 112.8 \quad (42)$$

where C_1 is the sum of the individual component complexities, C_2 is the complexity of each pairwise interaction, and C_3 represents the system's topological complexity. C_1 was estimated using subjective probability distribution techniques based on subject matter experts' judgment [43].

4.2 Test Case II: Change of Requirements

In this test case, the proposed complexity measure's ability to detect system design changes was examined. For this purpose, two new system requirements were introduced:

1) The customer shall customize his/her preferences (i.e., language and currency).

2) The Checking Account shall permit overdrafts.

The BDD shown in Figure 10 reflects the effects (circled in red) of the new requirements on the Customer and Checking Account components (e.g., preferences and overdraft, respectively).

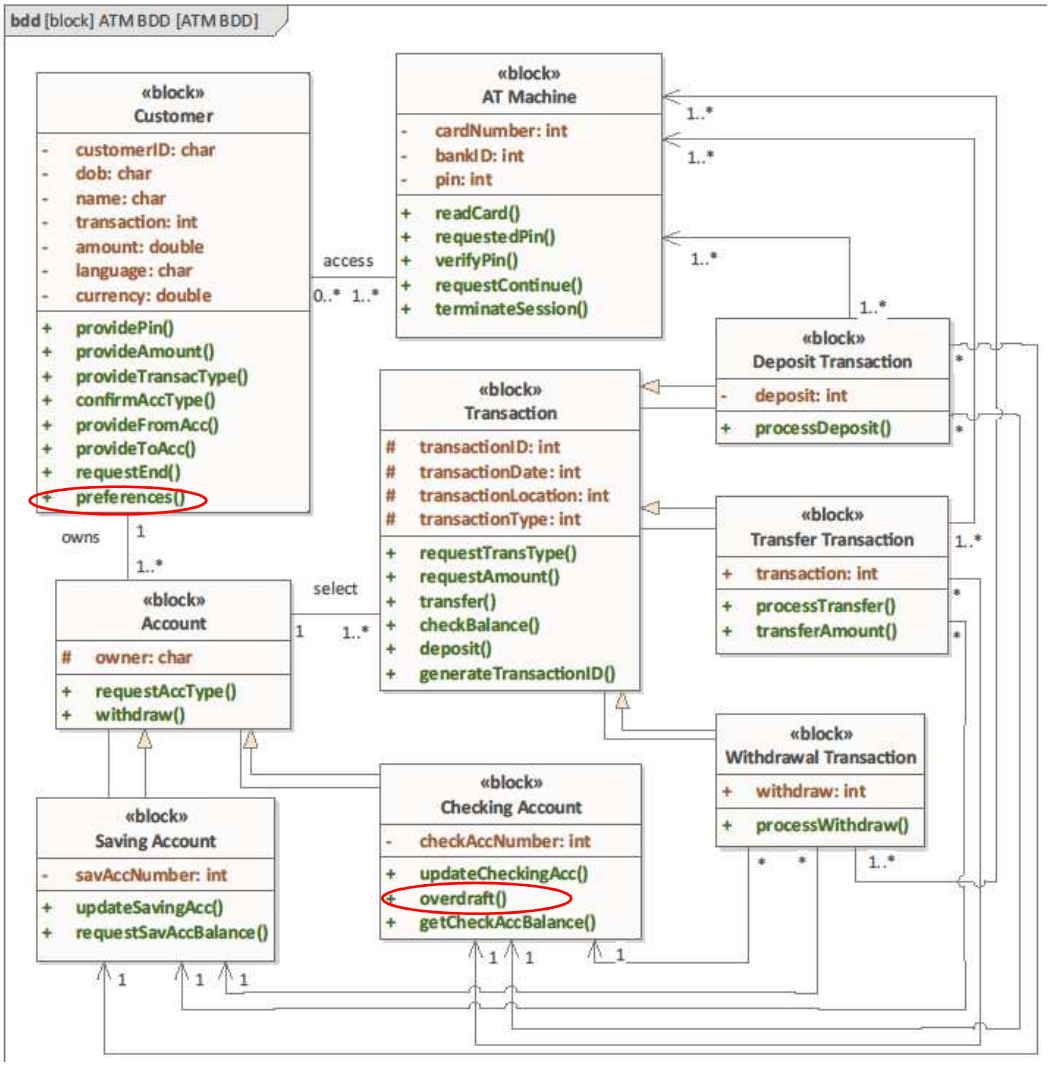

Figure 10. ATM BDD, including new requirements (©2021 IEEE).

An essential effect of the first requirement is the origination of a new system state: Customization, reflected in the new SMD of Figure 11.

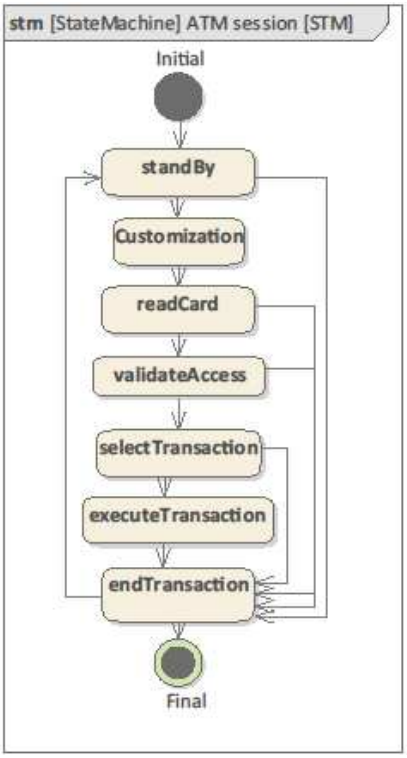

Figure 11. Modified ATM SMD with new requirements (©2021 IEEE)

Then, the new state is added to (37) as the 7th state. Assuming an arbitrary $x_7 = 15$, then the new complexity measure is expressed as:

$$C = \sum_{i=1}^{n=7} \mu_{Trans_type} |P_i \psi_i|^2 = \mu_{Trans_type} \sum_{i=1}^{n=7} |P_i \psi_i|^2 = \tag{43}$$

$$\mu_{Trans_type} \{ |0.01 * 1|^2 + |0.56 * 1|^2 + |0.15 * 1|^2 + |0.98 * 1|^2 +$$

$$|0.44 * 1|^2 + |0.01 * 1|^2 + |0.15 * 1|^2 \} = \mu_{Trans_type} * 1.513$$

The original Withdrawal's SD (Figure 5) had to be modified according to the new requirement changes. The new Withdrawal SD is shown in Figure 12, where the preferences operation (or method) was included in Customization, and the overdraft operation was included in executeTransaction. Similar steps to those applied in the obtention of the SD in Figure 12 can be followed to obtain the Deposit and Transfer

Transaction SDs, which are shown in Figure 13 and Figure 14, respectively (circled in red). Note that per requirements, the overdraft is included only in the Withdrawal SD, and the Deposit and Transfer Transaction SDs only include the preferences operation.

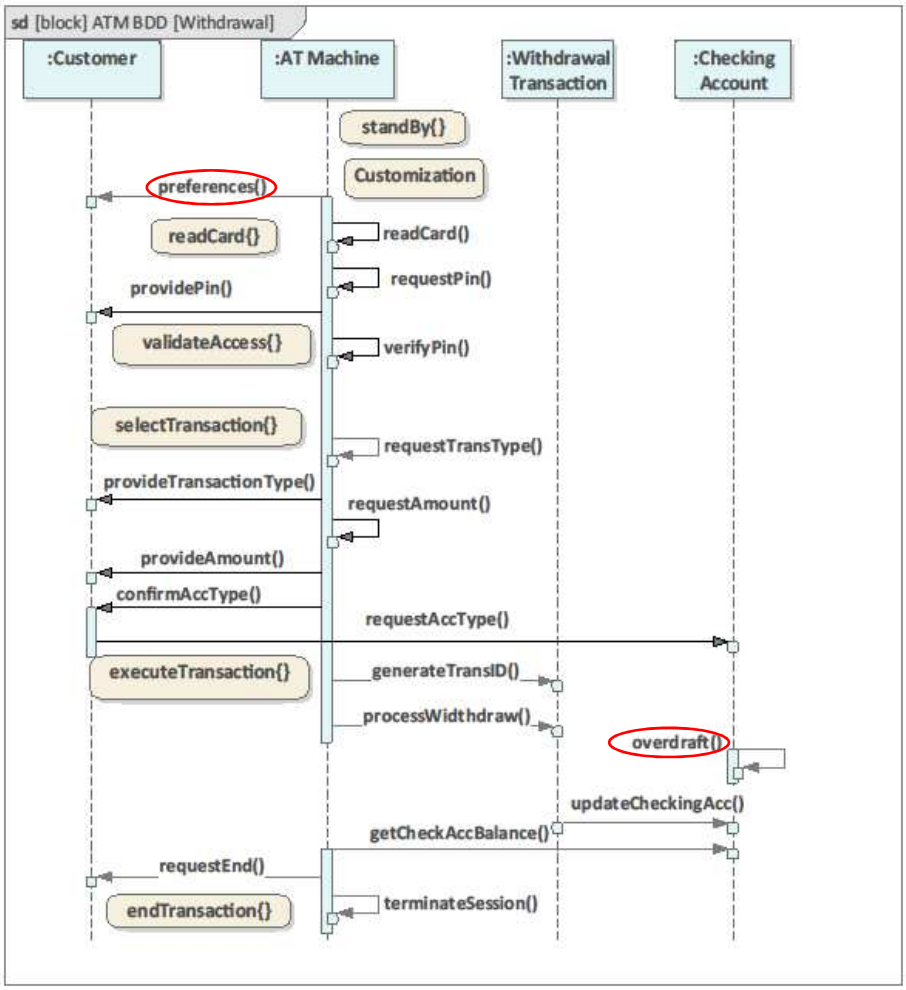

Figure 12. ATM SD's Withdrawal from the Checking Account with new requirements (©2021 IEEE).

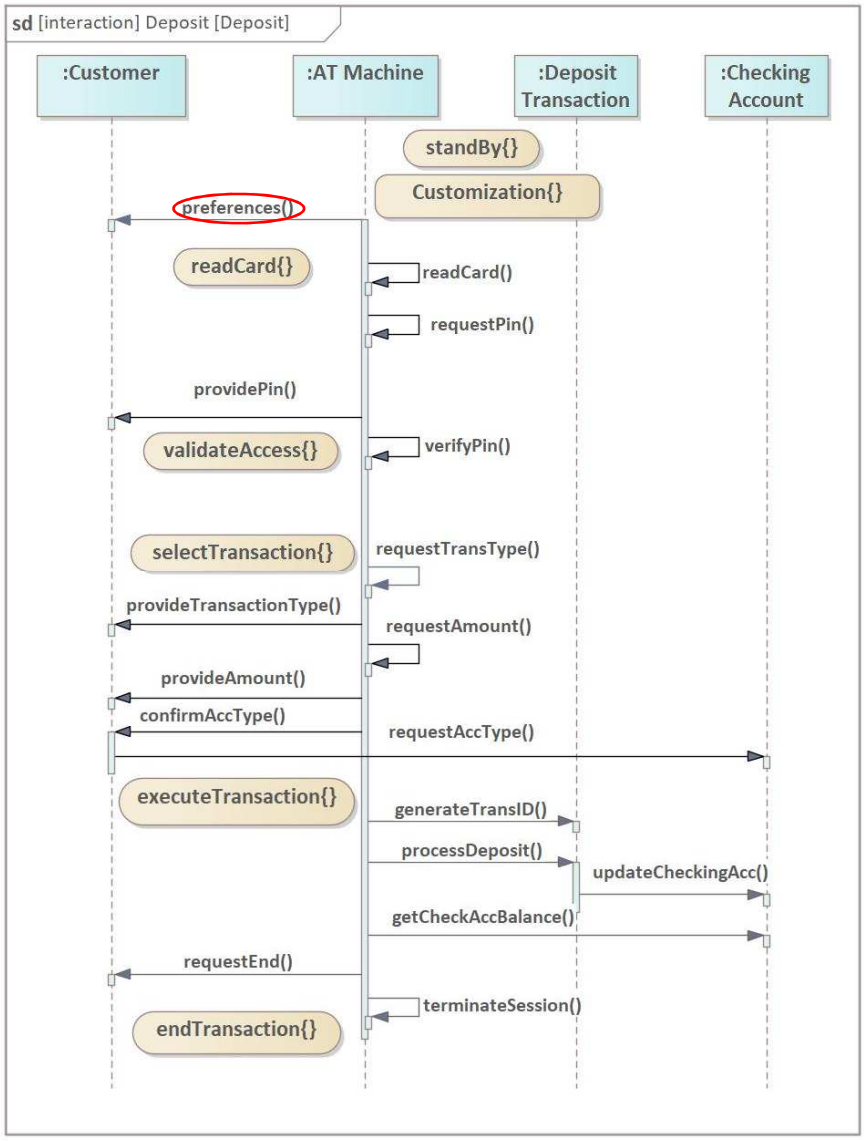

Figure 13. ATM SD's Deposit from the Checking Account with new requirements.

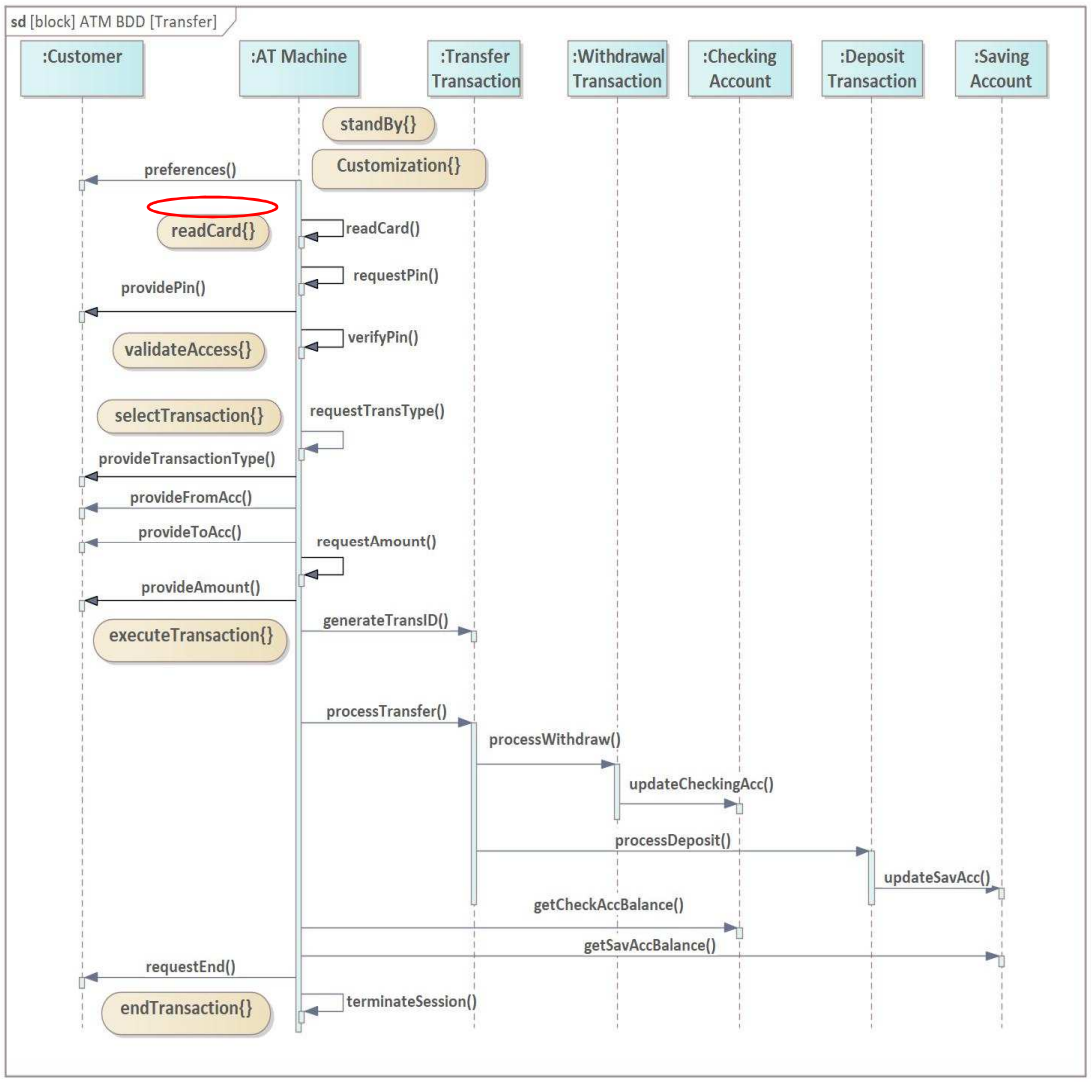

Figure 14. ATM SD's Transfer from the Checking Account with new requirements.

Following the same process as in Test case-I, the number of functions or methods of the i^{th} component, block, or class associated with a state can be calculated directly from the Withdrawal, Deposit, and Transfer operations' BDDs of Figure 12, Figure 13, and Figure 14, respectively. The number of methods or functions of each component during these operations are summarized in Table 4, Table 5, and Table 6.

Item	Component	Number of functions
1	Customer	6
2	AT Machine	6
3	Checking Account	4
4	Withdrawal Transaction	2
	Total number of functions $\mu_{Withdrawal}$	18

Table 4. Number of Functions per Component (Withdrawal - New requirements)

When the new requirements are included, the calculation of the system complexity according to (37) (Withdrawal operation from the Checking Account) is:

$$C_{withdrawal_new} = \mu_{Withdrawal} \sum_{i=1}^{n} |P_i \psi_i|^2 = 18 * 1.513 = 27.2 \quad (44)$$

Item	Component	Number of functions
1	Customer	6
2	AT Machine	6
3	Checking Account	3
4	Deposit Transaction	2
	Total number of functions $\mu_{Deposit}$	17

Table 5. Number of Functions per Component (Deposit– New requirements)

Similarly, when the new requirements are included, the calculation of the system complexity according to (37) (Deposit operation to the Checking Account) is:

$$C_{Deposit_new} = \mu_{Deposit} \sum_{i=1}^{n} |P_i \psi_i|^2 = 17 * 1.513 = 25.7 \quad (45)$$

Item	Component	Number of functions
1	Customer	7
2	AT Machine	6
3	Transfer Transaction	2
4	Withdrawal Transaction	1
5	Checking Account	2
6	Deposit Transaction t	1
7	Saving Account	2
	Total number of functions $\mu_{Transfer}$	21

Table 6. Number of Functions per Component (Transfer – New requirements)

Once again, when the new requirements are included, the calculation of the system complexity according to (26) (Transaction operation from the Checking Account) is:

$$C_{Transfer_new} = \mu_{Transfer} \sum_{i=1}^{n} |P_i \psi_i|^2 = 21 * 1.513 = 31.8 \quad (46)$$

Then, the result obtained in (41) is applied once again to obtain:

$$C_{Sys} = \sum_{1=1}^{m} \{C_{1,} C_{2,\ldots,} C_m\} = 2 * \{C_{Withdrawal_{new}} + C_{Deposit_{new}} + C_{Transfer_{new}}\} \quad (47)$$

$$= 2 * \{27.2 + 25.7 + 31.8\} = 169.4$$

Reevaluating the modified model with Sinha and de Weck's structural measure [10] in (8), the resulting system structural complexity is:

$$C_{struct} = C_1 + (C_2 * C_3) = 21 + (84 * 1.1) = 112.8 \quad (48)$$

The identical result obtained in (42) and (48) demonstrates the structural complexity measure's insensitivity to both the functional and behavioral changes (i.e., $\Delta C_{struct} = 0$). The structural complexity measure reported the same system complexity value in both cases because the new requirements did not entail any structural changes, as is confirmed by comparing the BDDs before and after the model modifications (See Figure 8 and Figure 10).

Contrasting the results obtained with the structural complexity measure, in the results from the comprehensive complexity measure, a comparison between (41) and (47) shows an increased system complexity of ΔCsys = 169.4 – 154.8 = 14.6 (i.e., a 9.4% increase in complexity). The result reported by the new complexity measure confirms the third hypothesis stated in Section 1.6: "A complexity measure that accounts for system

behavior, states, structure, and function is more sensitive to complexity changes than traditional complexity measures, which only account for structure or function." The proposed complexity measure is indeed accounting for system structure, like many traditional complexity measures. However, the new complexity measure is more advantageous than the traditional measures because it also accounts for function, states, and behavioral changes in the system, which is in line with the additional system capabilities triggered by the new requirements. The formulation of the proposed complexity measure includes a term to calculate functional complexity involving the number of methods (or functions) weighted by their corresponding probabilities, as described in (12)-(21). This term enables the discovery of the most complex elements in the system and indirectly accounts for structural complexity. When the model's structure (i.e., components, interconnections, or configuration) is subjected to modifications, the corresponding functions are also affected, which is sensed by the complexity measure.

4.3 Summary

The results in this chapter endorse the hypotheses formulated to test the research questions stated in Sections 1.5 and 1.6.

Hypothesis 1: "System behavior is a quantifiable system property correlated to system complexity." This hypothesis was formulated to answer the question, "Is system behavior a quantifiable magnitude, and how is it related to system complexity?" The mathematical equations that show the correlation between system complexity and system behavior were formulated in Chapter 3, where the mathematical formulation was developed. Equations (19) to (28) describe complexity as a function of changes in system state, which determine the behavior of a system. The correlation between system

behavior and system complexity was verified in this chapter by applying the mathematical formulation to an ATM system model developed in Section 3.1.3. The quantification of this relationship between system complexity and behavior was evidenced in (33) to (48). Consequently, the achievement of Research Objective 1 devised in Section 1.5, "Formulate a mathematical model of a complexity measure based on a system structure, function, and behavior approach," was completed. The mathematical model integrated system structure, function, and behavior holistically and was effectively applied to the ATM system model. The mathematical model proved sufficient to solve the problem of quantifying system complexity.

Hypothesis 2: "A resulting system model construct contains stored information that can be retrieved and quantified by a comprehensive complexity measure." This hypothesis was formulated to answer the question, "How can a holistic complexity measure be integrated with a model of a system?" This hypothesis implied that there was a link between the mathematical model and the system model. An MBSE approach was used to construct a model of an ATM system in Chapter 3. In this chapter, the quantification of the system model complexity was performed by using the new complexity metric. The quantification exercise revealed the synergy between the mathematical model and the system model is evidenced in Figure 5 to Figure 14 and Table 1 to Table 6 presented in this chapter. Consequently, Research Objective 2 devised in Section 1.5, "Develop a system model using model-based methods and tools and integrate it with the novel comprehensive complexity measure," was achieved.

Hypothesis 3: "A complexity measure that accounts for system behavior, states, structure, and function is more sensitive to complexity changes than traditional

complexity measures, which only account for structure or function." This hypothesis was formulated to answer the question, "Is a complexity measure based on a system states and behavior approach more advantageous in assessing system complexity than traditional complexity measures?" The results obtained in (41) and (47), and (42) and (48) demonstrated the veracity of this hypothesis. Consequently, Research Objective 3 devised in Section 1.5, "Determine if a comprehensive complexity measure approach provides advantages over traditional approaches in measuring complexity," was met. This objective was accomplished in this chapter. The results confirmed that the new complexity measure is sensitive not only to structural changes but also to functional and behavioral changes occurring in the model. In contrast, traditional structural measures cannot detect these changes because they are usually designed to quantify either functional or structural changes, but not both simultaneously.

Although the test cases were chosen from the SW arena, the proposed complexity measure can be applied to HW or integrated SW and HW systems. The proposed complexity measure and methodology illustrated in the test cases can also be applied to any system, regardless of its structural or functional size.

Chapter 5: Conclusions, Limitations, and Recommendations

This chapter discusses the research findings, accomplishments, limitations, and recommendations for future work.

5.1 Conclusions

In this research, a novel comprehensive measure was proposed for quantifying the complexity of engineered systems. The proposed complexity measure formulated an engineering system's complexity as a single quantifiable system property that incorporates structure, function, states, and behavior. The proposed measure derived its mathematical formulation from MQM and signal processing theory. The measure was also integrated with MBSE to identify components (blocks or classes) that constitute the primary sources of complexity. The new complexity measure was intended for real-world systems applications; therefore, this research performed two test cases, based on an ATM system, to demonstrate a method for that extensible application of the measure.

In contrast to the numerous complexity measures used today, which account only for either system structure or functionality, the proposed measure simultaneously accounts for system structure, functionality, and behavior. When the model's structure (i.e., components, interconnections, or configuration) is subjected to modifications, the corresponding functions are also affected, which is sensed by the proposed complexity measure.

Because the new complexity measure is sensitive to structure, function, states, and behavior, which all go hand in hand, it provides a more comprehensive understanding of system complexity. The proposed measure allows the identification of the more complex system elements, which represent the heft of the development effort.

The proposed complexity measure provides system engineering practitioners with a powerful tool to measure the system complexity of a single system design or different designs, facilitate design decisions, implement complexity management [44], and identify and prevent possible unwanted system behaviors [45]. Therefore, system engineering professionals should use the resulting complexity information early in the design process to focus on those heavily complex elements and generate more accurate schedule and cost estimates.

Conventional estimation techniques and management practices rely on historical data and do not account for configuration uncertainty and incomplete requirements, often underestimating the system complexity and leading to inaccurate estimates. The proposed complexity measure does not depend on historical data and addresses uncertainty by using probabilities. In the new complexity measure, an element's individual functional complexity is defined as the probability of successfully fulfilling a requirement, which relates functional complexity to the level of functional information content. Simple probabilistic models can be initially assumed when no historical data is available. Given the modeling method's iterative nature, the probabilities can be refined as more information is learned and the design matures.

Modeling and architecting a complex system are generally time-consuming and intensive activities, particularly for new systems. The comprehensive complexity measure is flexible and allows for simplifications that may result handy to alleviate the burden of these activities (see (19) and (20)).

The proposed complexity measure is dynamic because it is grounded on system states, which by definition, fully define the system behavior at some point in time

(dynamism). Ultimately, the system's behavior is determined by system state changes due to the dynamic relationships between components and functions. Components, interactions, configuration, functions, and states are closely interdependent, transforming and evolving mutually. In complex engineering systems, behavior changes are triggered by requirement modifications that may entail state conditions, structure, and functionality changes. Interactions are not static and may change over time in the short or long term. For that reason, the proposed complexity measure coupled states and interactions stochastically rather than deterministically to sense any changes in structure, functionality, or behavior. The results demonstrated the advantages of using the proposed complexity measure over other complexity measures because it considers the system as a whole, detecting structural, functional, and behavioral changes simultaneously.

5.2 Limitations

This work is limited to complex engineered systems. Self-organizing systems are out of the scope of this research. Self-organizing systems do not need an internal or external controller or agent to exhibit an organized behavior, such as swarm robots and many biological or social systems. This research is concerned with complex engineered systems that need an architect, designer, or other systems engineering professional to trigger and implement system design changes according to customer requirements.

The synergy between the proposed complexity measure and the system model is a mighty advantage, but it may be a limitation. Indeed, the more comprehensive the complexity measure, the better the understanding of the system complexity. However, the complexity measure evaluates what the model can capture from the system. Therefore, the complexity measure is dependent on the model's accuracy and completeness. Finding

the appropriate level of detail (completeness of specificity) for a complex system model is generally not a small task. The modeling activity demands a significant degree of experience and creativity from a designer. Many potential models can describe a system, and a multi-view model or several models may be necessary to understand and characterize the system. In practice, the designer usually begins the system modeling at a certain degree of detail and finishes it at some other degree of detail characterized by a balance between accuracy and completeness of specificity.

Furthermore, the formulation of the complexity measure is limited to steady-state. The proposed complexity measure is designed for systems with state stability at a given time for a moderately long period. In principle, the proposed complexity measure does not have enough resolution for accounting transient states in highly dynamic engineering systems. This limitation may not be very stringent because most SOI considered during the design process have a fixed set of states which are not expected to change at a full tilt.

This research does not claim that the comprehensive complexity measure is standard or completely universal. Currently, there is not a universal or "standard" complex system. A standard complex system is interpreted here as a system that is well-established and widely accepted by the systems engineering community by exhibiting properties and behavior that are repeatable and reproducible. A standard system would always present a defined and constant complexity value, as the meter in the modern metric system (known as the International Systems of Units) is always the same length. There is no equivalent of the meter stick in measuring system complexity. Like any other complexity measure in use, such as the ones reviewed in Chapter 2, the proposed complexity measure provides measurement values that are relative to each other. A direct

comparison of two complexity measures is technically incorrect because of the futility of obtaining a standard and consistent scale of system complexity. In the absence of such a standard, a comparison between two complexity measures needs to be done in two stages. One stage is needed for determining a system complexity baseline, and another one for reassessing the system complexity after introducing system modifications. For this reason, the demonstration of the benefits of applying the proposed complexity measure required the two test cases.

Finally, as defined in equation (9), the system behavior is described in terms of a linear combination of basis functions weighted individually by a probability that the system is in a particular state. The probabilistic approach to describe the system's dynamism accounts for uncertainty, generally an innate characteristic of any engineering project. Although this probabilistic approach provides a good solution against uncertainty, it also evidences a lack of knowledge on the system behavior. The probabilistic description is intended to counterbalance the formulation's inability to deterministically forecast states and interactions, which can be interpreted as an unavoidable consequence of the formulation's quantum mechanics roots.

5.3 Recommendations for Future Research

Despite the energy devoted to this work, there is room to explore some ideas in greater detail. The following topics represent opportunities for future research.

Future research may include a performance study to establish correlations between requirements, "ilities,' and system complexity (measured with the proposed system complexity measure), to understand their inter-dynamics and effects on system behavior. The system complexity measure and associated methodology proposed in this research

constitute precious resources for systems engineers, as they quantify system complexity, identify complex elements in the system, and gain a more thorough understanding of a particular system's complexity. Complex elements and their interfaces are critical in a system but carry a higher risk if they fail during operation due to disruption, decay, breakage, or unpredictable behavior. However, even after applying risk retirement strategies during development to cope with complexity, complex elements may persist. Operating characteristics known as "ilities" (such as reliability, maintainability, supportability, usability, producibility, accessibility, and disposability) should be considered during the design and development process. To guarantee successful system operation, requirements involving these "ilities" should be well-defined and understood at the design phase. However, in complex systems, the "textual" interpretation of requirements is not always straightforward, and the complete set of implications in terms of added complexity to the system cannot always be anticipated. Establishing correlations between a comprehensive system complexity assessment and technical performance measures can help to determine design margins for the technical requirements being tracked. Those margins can provide helpful insights to apply in technical trade-offs for a design or comparative designs, reduce overall system risk, and fine-tune project forecasts.

The above research recommendation is also tied to product cost. Future research may include studies to track the accuracy of product cost forecasts based on the proposed system behavior-based complexity measure. Complex components impose challenges on engineers, who constantly have to work with budget constraints, because these complex components may require more expensive materials and processes, intensive person-hours, and costly life-cycle "ilities" such as maintainability, serviceability, and disposability.

Experience suggests that the decisions made during the conceptual design and early planning regarding system complexity significantly affect the estimated SOI life-cycle cost. However, project forecasts involving these costs are theoretical at the design phase of the system life-cycle. When system production starts, the project forecast has to prove whether it is correct in terms of schedule, cost, and profit.

Furthermore, a sensitivity study can be carried out to uncover important terms or complexity allocation among elements when comparing several products or design scenarios. After evaluating the complexity of an SOI and inspecting its model, complex elements can be identified. Further examination of these elements may lead to the discovery of possible input factors that originate complexity, such as flawed assumptions, missing requirements and constraints, incomplete definitions, and insufficient data input. The identification of where the complexity resides in an SOI is an important starting point. However, some additional questions need to be addressed, such as how sensitive the complexity results are to variations in these ambiguous or incomplete input factors. These questions can be answered by identifying the elements that mainly contribute to system complexity, determining the burning input factors that directly affect complexity, reflecting the variation of these factors in the model, and applying the new complexity measure to determine the impact.

Finally, a more general formulation of the complexity measure can be found through a study incorporating transient states. Incorporating transients may allow the analysis of highly dynamic complex systems. Introducing perturbations to a system, such as system failures to respond to certain stimuli or conditions, may lead to system instability and unexpected system behavior. A more general formulation of the

comprehensive complexity measure enables assessing the system complexity and observing the system behavior during different periods. The characterization of the system under those conditions may play an essential role in determining system resilience.

References

[1] SEBoK Editorial Board, *Guide to the Systems Engineering Body of Knowledge (SEBoK)*, v.2.1, Hoboken, NJ, 2020.

[2] F. Ameri, J. D. Summers, G. M. Mocko, and M. Porter, "Engineering design complexity: An investigation of methods and measures," Res. Eng. Des., vol. 19, no. 2–3, pp. 161–179, 2008.

[3] W. Weaver, "Science and Complexity," American Science, v36, pp: 536-544, 1948.

[4] J. R. Paiva et al., "Metric for Calculation of System Complexity based on its Connections," Trans. Environ. Electr. Eng., vol. 2, no. 1, 2017.

[5] A.V. Deshmukh, J. J Talavage; M. M. Barash, "Complexity in manufacturing systems, part 1: Analysis of static complexity," IIE Trans., vol. 30, no. 7, 1998.

[6] F. Salimi and F. Salimi, "A Systems Approach to Managing the Complexities of Process Industries," 1st ed., Elsevier, 2018.

[7] S. Huang, G. Wang, X. Shang, and Y. Yan, "Reconfiguration point decision method based on dynamic complexity for reconfigurable manufacturing system (RMS)," J. Intell. Manuf., vol. 29, no. 5, pp. 1031–1043, 2018.

[8] D. Braha and O. Maimon, "The Measurement of a Design Structural and Functional Complexity," IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans, vol. 28, no. 4, pp. 527–535, 1998.

[9] H. A. Bashir and V. Thomson, "Estimating design complexity," J. Eng. Des., vol. 10, no. 3, pp. 247–257, 1999.

[10] K. Sinha and O. L. de Weck, “Empirical Validation of Structural Complexity Metric and Complexity Management for Engineering Systems,” Syst. Eng., vol. 19, pp. 193–206, 2016.

[11] A. Pugliese and R. Nilchiani, “Developing Spectral Struct. Complexity Metrics,” IEEE Syst. J. vol. 13, no. 4, pp. 3619–3626, 2019.

[12] J. Fischi, R. Nilchiani, and J. Wade, “Dynamic Complexity Measures for Use in Complexity-Based System Design,” IEEE Syst., vol. 11, no. 4, pp. 2018-2027, 2017.

[13] N. P. Suh, “Designing-in of Quality Through Axiomatic Design,” IEEE Trans. Reliab., vol. 44, no. 2, pp. 256–264, 1995.

[14] D. Dori, *Object-Process Methodology: A Holistic System Paradigm*. New York, NY, USA, Springer, 2002.

[15] E. Crawley, B. Cameron, and D. Selva, *Systems Architecture: Strategy and Product Dev. for Complex Systems*. Pearson, 2015.

[16] M. E. J. Newman, “The structure and function of complex networks,” SIAM Rev., vol. 45, No 2, pp. 167–256, 2003.

[17] J. L. Mathieson, “Connective Complexity Methods for Analysis and Prediction in Engineering Design,” MS thesis, Mech., Eng. Dept., Clemson University, 2011.

[18] R. Quashem, “Design Structure Matrix: Models, Applications and Data Exchange Format,” MS Thesis, Univ. of Lethbridge, Dept of Math and Comp. Science, Canada, 2015.

[19] J. D. Summers and J. J. Shah, “Developing Measures of Complexity for Eng. Design,” J. Mech. Des., vol. 132, no. 2, p. 021004, 2010.

[20] S. A. Sheard and A. Mostashari, “Principles of Complex Systems for Systems Engineering,” Syst. Eng., vol. 12, no. 4, pp. 295–311, 2008.

[21] S. A. Sheard and A. Mostashari, “A Complexity Typology for Systems Engineering,” INCOSE Int. Symp., vol. 20, no. 1, pp. 933–945, 2010.

[22] B.S, Blanchard and W.J. Fabrycky, *Systems Engineering and Analysis.* 5th ed. Prentice-Hall International, 2011.

[23] E. Rechtin, *Systems Architecting–Creating & Building Complex Systems*. Prentice Hall PTR, 1991.

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

[25] B. Douglass, *Agile Systems Engineering*.1st ed. Elsevier, 2016.

[26] INCOSE, *Systems Engineering Handbook: A Guide for System Life-cycle Processes and Activities*. 4th. Ed., San Diego, CA, USA, 2015.

[27] S. Friedenthal et al., *A Practical Guide to SysML. The Systems Modeling Language*. 3rd ed., MK/OMG Press, 2015.

[28] A. M. Madni and M. Sievers, “Model-based systems engineering: Motivation, current status, and research opportunities,” Syst. Eng., vol. 21, no.2013, pp. 172–190, 2018.

[29] A. Sage and W. Rouse, *Handbook of Systems Engineering and Management*. John Wiley & Sons, 2014.

[30] Eugene P. Wigner, “Annals of Mathematics Characteristic Vectors of Bordered Matrices with Infinite Dimensions,” Ann. Math., vol. 62, no. 3, pp. 548–564, 1955.

[31] P. Atkins and R. Friedman, *Molecular Quantum Mechanics*. 4th ed., Oxford University Press, 2005.

[32] X. Li, et al., "G*raph Energy*. 1st ed., Springer-Verlag New York, 2012

[33] J. A. Bondy, and U. S. R. Murty, *Graph Theory with Applications*. 5th ed. Elsevier Science Publishing Co., Inc, 1982.

[34] G. Pinski and F. Narin, "Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics," Inf. Process. Manag., vol. 12, no. 5, pp. 297–312, 1976.

[35] Method for node ranking in a linked database, by Lawrence Page (2001, Sep 4). Patent US6285999 B1.

[36] E. Kreyszig, *Advanced Engineering Mathematics*. 10th ed., Willey, 2011.

[37] L. E. Ballentine, "The statistical interpretation of quantum mechanics," Rev. Mod. Phys., vol. 42, no. 4, pp. 358–381, 1970.

[38] M. Marcus and H. Minc, "A Survey of Matrix Theory and Matrix Inequalities," rev. ed., New York, Dover, 1992.

[39] M. Maya-Mendieta, et al., "Hermitian operators and boundary conditions," Rev. Mex. Fis., vol. 58, no. 1, pp. 94–103, 2012.

[40] R. Frigyes and Sz-Nagy, *Functional analysis*. 2nd Ed., Dover Publications, New York, 1990.

[41] A. Oppenheim and R. Schafer, *Digital Signal Processing*. 1st ed., Pearson, 1975.

[42] U. Kuehn, *Integrated Cost and Schedule Control in Project Management*. 2nd ed., Management Concepts, 2011.

[43] P. Garvey, *Probability Methods for Cost Uncertainty Analysis-A Systems Engineering Perspective*. CRC Press, NY, 2000.

[44] B. Murray et al., "META II: Complex Systems Design and Analysis (CODA)," United Technologies Co. - AFRL Propulsion Directorate, Air Force, OH, USA, Rep. AFRL-RZ-WP-TR-2011-2102, Aug. 2011.

[45] B. Beihoff et al., "A World in Motion: SE Vision 2025," INCOSE, pp. 1–9, 2014.

[46] S. Eppinger and T. Browning, *Design Structure Matrix Methods and Applications*. Cambridge, Mass: MIT Press, 2012.

[47] F. Bonilla, T. Holzer, and S. Sarkani, "Complexity Measure for Engineering Systems Incorporating System States and Behavior," IEEE Systems Journal, DOI: 10.1109/JSYST.2020.3033792 (Early access).

[48] "International Standard ISO / IEC / IEEE Systems and Software Engineering – System Life-cycle Processes." ISO/IEC/IEEE 15288:2015.

[49] C. Shannon, *A Mathematical Theory of Communication.* Bell System Technical Journal, vol. 27, no. 3, pp. 379–423, 1948.

[50] Babuscia, Alessandra, et al. "Statistical Risk Estimation for Communication System Design: Development of Optimization Frameworks." IEEE Aerospace Conference Proceedings, vol. 7, no. 1, pp. 125–36 IEEE, 2013.

[51] Yoshida, Justin, et al. "Complexity Analysis of the Cost-Effectiveness of PI-Led NASA Science Missions." IEEE Aerospace Conference Proceedings, edited by IEEE, pp. 1–14, 2013.

[52] G. Filippazzo, “Complexity Based Cost Estimating Relationships for Space Systems.” IEEE Aerospace Conference Proceedings, vol. 6, IEEE, pp. 4093–98, 2004.

[53] K. Sinha et al. “Structural Complexity: Quantification, Validation and Its Systemic Implications for Engineered Complex Systems.” Proceedings of the International Conference on Engineering Design, ICED, vol. 4 DS75-04, pp. 189–98, 2013.

[54] E. Weyuker, “Evaluating Software Complexity Measures.” IEEE Transactions on Software Engineering, vol. 14, no. 9, pp. 1357–65, 1988.

[55] N. Azizian, et al. “A Framework for Evaluating Technology Readiness, System Quality, and Program Performance of US DoD Acquisitions.” Systems Engineering, vol. 14, no. 3, pp. 305–26, 2010.

[56] A. Olechowski, et al. “Technology Readiness Levels: Shortcomings and Improvement Opportunities.” Systems Engineering, vol. 23, no. 4, pp. 395–408, 2020.

[57] A. Pugliese, et al. “Acquisition and Development Programs Through the Lens of System Complexity.” Proc. Annu. Acquisition Res. Symp. Naval Postgraduate School, pp. 136–54, 2018

[58] L. M. Laird and C. Brennan, *Software Measurement and Estimation: A Practical Approach*. Hoboken, NJ: Wiley, 2006.

[59] J. Tranquillo, *An Introduction to Complex Systems. Making Sense of Changing World*. Springer Nature Switzerland, 2019.

[60] M. Mitchell, *Complexity, A Guided Tour*, Oxford University Press, 2009.

[61] Bill Haskins, et al., “Error Cost Escalation Through the Project Life-cycle.” INCOSE International Symposium, pp. 1723–37, Wiley, 2004

[62] H. Sayama, *Introduction to the Modeling and Analysis of Complex Systems.* Bingamon University, Open SUNY Textbooks, Geneseo, NY, 2015.

[63] Deyst, John, and Karen Willcox. “An Information-Theoretic Metric of System Complexity With Application to Engineering System Design.” Journal of Mechanical Design, vol. 134. October, 2012.

[64] R. Sonntag and G. Van Wylen, *Fundamentals of Thermodynamics*. John Wiley & Sons, 2019.

[65] E. Basem and K, Yang. “The Components of Complexity in Engineering Design.” IIE Transactions, vol. 31, no. 10, 1999, pp. 925–34.

[66] J. Mathieson, and Joshua D. Summers. “A Protocol for Modeling and Tracking Engineering Design Process through Structural Complexity Metrics Applied against Communication Networks.” Concurrent Engineering Research and Applications, vol. 25, no. 2, pp. 108–22, 2017.

[67] D. Braha and O. Maimon, “The Design Process: Properties, Paradigms, and Structure,” IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans., vol. 27, no. 2, pp. 146–66, 1997.

[68] Chen, Li, and Simon Li, “Analysis of Decomposability and Complexity for Design Problems in the Context of Decomposition,” Journal of Mechanical Design, Transactions of the ASME, vol. 127, no. 4, pp. 545–57, 2005.

[69] B. Zhou and I. Gutman, "Nordhaus-Gaddum-Type Relations for the Energy and Laplacian Energy of Graphs," Vol. 32, no. 32, Serbian Academy of Sciences and Arts, 2007 Bulletin, pp. 1–11.

[70] I. Gutman and B. Zhou, "Laplacian Energy of a Graph," Linear Algebra and Its Applications, vol. 414, no. 1, pp. 29–37, 2006.

[71] J. Chemiavsky and C. Smith, "On Weyuker's Axioms for Software Complexity Measures," IEEE Transactions on Software Engineering, vol. 17, no. 6, pp. 636–39, 1991.

[72] J. Wu, M. Barahona, Y. Tan, and H. Deng, "Robustness in Complex Networks," IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans, vol. 41, no. 6, pp. 1244–1252, 2011.

[73] G. Smith, *Essential Statistics, Regression, and Econometrics*, 1st Ed., Elsevier, 2012.

[74] A. Papoulis, *Probability Random Variables, and Stochastic Processes*, 3rd ed., McGraw-Hill, 1991.

[75] R. L. Ott and M. Longnecker, *An Introduction to Statistical Methods and Data Analysis*, 7th Ed., Cengage Learning, 2016.

[76] R. R. Clements, "The development of methodologies of mathematical modelling," Teach. Math. its Appl., vol. 1, no. 3, pp. 125–131, 1982.

[77] J. Banks and J. S. Carson II, *Discrete Event System Simulation*, 1st ed., Prentice-Hall Inc., 1984.