Mathematical Foundations for Validation in Systems Engineering

Authors:
Hanumanthrao Kannan* (corresponding author)
Assistant Professor
Grado Department of Industrial and Systems Engineering, 225 Durham Hall, Virginia Tech, Blacksburg VA 24061
INCOSE regular member
Email: hkannan@vt.edu, Phone: 540-231-1839

Mayuranath SureshKumar
Graduate Research Assistant
Grado Department of Industrial and Systems Engineering, 225 Durham Hall, Virginia Tech, Blacksburg VA 24061
Email: mayursk@vt.edu

Funding: This work did not receive any funding.

Mathematical Foundations for Validation in Systems Engineering

Hanumanthrao Kannan, and Mayuranath SureshKumar
Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, Virginia, USA

Abstract

The primary goal of Systems Engineering is to develop a solution that best satisfies the needs of stakeholders. Validation is the process of checking if the developed solution satisfies the needs of the stakeholders. Current validation practices are heuristics-based and lack mathematical foundations, which may lead to suboptimal validation strategies. To ensure the correct and accurate validity of the system of interest, several artifacts throughout the system lifecycle must be considered, including but not limited to stakeholders' needs, requirements, design, verification, etc. This paper characterizes validation in terms of these artifacts by proposing novel theoretical insights using Propositional logic as the foundation. The mechanism of representation for stakeholders' needs plays a significant role in performing validation. This paper discusses the challenges associated with using existing textual stakeholder needs and preference functions with respect to validation in relation to the theoretical insights proposed. In addition, this paper highlights how the previously developed Modal preference logic can be used as an effective mechanism to develop a normative approach to validation in Systems Engineering.

I. MOTIVATION

The primary goal of Systems Engineering (SE) is to develop a solution that best satisfies the needs of the stakeholders [1, 2]. Validation is the process by which systems engineers determine whether the solution satisfies the needs of the stakeholders [1, 2]. In other words, validation helps to answer the question: *Did we build the right system?* According to ISO 9000:2005, validation is a collection of processes that ensures and provides assurance that a system can achieve its intended usage, goals, and objectives (i.e., meet stakeholder needs) in its intended operational environment.

In SE, system validation is the process of checking whether the final system of interest (SOI) has met all the stakeholders' needs. This involves performing validation activities in the form of inspection, measurement, analysis, demonstration, test, etc., to check for the satisfaction of validation criteria that act as proxies for the stakeholders' needs. Based on the validation evidence, an agreement is reached between the stakeholders and the contracting team whether to rework or proceed with deployment. Although system validation depends on the SOI meeting stakeholder's needs, to ensure we are building the right system, it is necessary to provide evidence for the satisfaction of various artifacts, including but not limited to design and verification requirements, throughout the SE lifecycle. Here validating requirements entails ensuring that its content is justified and relevant to stakeholders' needs. Validating a system's design implies ensuring that it meets the requirements, where verification is used to check whether the design meets the requirements through verification activities that generate evidence on the verification criteria.

Current approaches to validation are not rigorous and are based on heuristics and best practices, and involve a significant amount of human judgment. This results in several challenges. First, there is no appropriate measure or unit to access validity, i.e., present validation methods often discuss how a system has high or low validity, without a meaningful measure for validity. Second, validating the requirements with respect to the stakeholders' needs is difficult since there are no rigorous processes for translating needs to requirements. Furthermore, requirements are derived from needs, which are typically created using heuristics, expert opinion, and historical facts. As a result, it is possible to have a system that may satisfy the needs but not the requirements, or vice versa. Third, there are acceptable practices/guidelines for translating needs to requirements, but there is lack of theoretical foundations that serve as a basis to claim

that a system is valid. Finally, there is a significant confusion between verification and validation, and the relationship between the two. Validation establishes that the product, service, or enterprise as delivered meets its intended purpose, whereas verification determines whether a product accurately reflects its requirements [1, 2]. An important question that needs to be addressed is whether a completely verified system implies a valid system, and vice versa.

Given all these challenges, the idea of identifying an optimal validation plan is out of question since the elements of validation are not treated formally. As highlighted by past NSF and DoD events [3-7], and the NASA and INCOSE SE consortium[8], it is critical to identify optimal validation procedures that ensure that the right system has been built without adding significant cost and time. Additionally, since validation of the solution with respect to the stakeholders' needs dictates the successful realization of a project, it is critical to treat validation mathematically. Ultimately, there is a need for a formal treatment of all the elements associated with system validation to enable (1) Novel theoretical foundations for validation in SE; and (2) Methodologies that enable optimal identification of validation plan.

This paper serves as a preliminary step towards addressing some of the above-mentioned challenges. Specifically, this paper focuses on the following:

i. A characterization of validation in SE by defining all the elements associated with validation.
ii. Theoretical insights on the problem of validation, and the relationship between the phases of validation, using Propositional logic.
iii. An extensive discussion of the challenges associated with using existing mechanisms for stakeholders' needs representation (textual needs, and preference functions) in relation to validation
iv. Modal preference logic as a normative framework to facilitate validation

The paper is organized as follows: Section II provides a comprehensive literature review concerning validation in a wide range of relevant domains. Section III provides necessary background of Propositional logic, which will be used in Section IV to propose novel foundations for validation in SE. This is followed by section V that discusses the challenges in validation associated with the existing mechanisms for representing stakeholders' needs in relation to the proposed foundations in Section IV. Section VI highlights future directions to overcome certain challenges in validation using the previously developed *Modal preference logic*. Section VII concludes the paper.

II. LITERATURE REVIEW

This section focuses on reviewing existing literature on validation in a few relevant domains including Modeling and Simulation, Design methods, and Software engineering.

A. Modeling and Simulation

Purpose: Validation in this domain is the process of assessing how accurate a simulation model and its accompanying data are in representing the real world from the perspective of the model's intended usage [9]. The validation process focuses on assessing if the disparities between observed behavior of system elements and corresponding elements of a simulation model are acceptable given the model's intended usage. If the disparities cause a reasonable agreement to not be reached, the model is revised to get it closer to the actual system's observed behavior (or inaccuracies in observation/experimentation or reference models/analyses are detected and corrected).

Procedure: There exists a variety of methods to validate the model [10, 11], namely, 1) Comparison to other models (i.e. the outputs of the model to be validated are compared to the known results of a similar valid model and this comparison is used to infer the validation of the model), 2) Face Validity (i.e. asking experts whether the model's input-output relationship and logic are correct or not), 3) Historical Data

Validation (i.e. When historical data is collected expressly for the purpose of building and testing a model, part of the data is used to create the model, while the rest is utilized to test if the model acts as the system intended), 4) Parameter Variability Sensitivity Analysis (i.e. It's a way for calculating how the uncertainty of one or more input variables can cause uncertainty in the output variables. This analysis is useful because it improves the model's prediction accuracy by evaluating qualitatively and quantitatively the model's response to changes in input variables. In other words, the expected values of various parameters can be used to assess the robustness or sensitivity of the results as a result of these changes and to determine the values beyond which the results change dramatically), 5) Predictive Validation (i.e. The model is used to predict (forecast) the behavior of the system, and then the behavior of the system is compared to the model's forecast to see if they are the same. The data for the system could originate from an operational system or from experimentation on the system, such as field tests)

Thus, the main procedures to be followed here are as follows, 1) Model Verification, Validation, and Accreditation (VV&A) plan development with quantitative model validation requirements, 2) Meticulous planning for the collection of validation data (if data-driven validation is needed) and 3) The formation of a working group comprised of domain experts and analysts. Although there are several methods for performing validation, there are also challenges that must be addressed. The following section discusses some of the common challenges to validating a model.

Challenges [12]: In the case of method 1 mentioned above, no matter how comparable a valid model is found, there are bound to exist certain differences that should be taken into consideration before carrying out validation. Second, with method 2, the expert's advice is always subjective, which creates a bias issue. It is difficult to design accurate test cases or trials to validate a model if the model taken into consideration is a relatively new concept without the existence of any prior knowledge on that particular field [13]. Since the majority of the methods discussed above are data-driven, caution and effort should be used when gathering data. Furthermore, gathering an all-encompassing data while considering every possible circumstance is quite difficult. Finally, complex systems, which are becoming more frequently the topic of modeling efforts, have specific distinguishing characteristics that make them more challenging to model and perform validation.

B. Design Methods

Purpose: Investigations into whether a suggested design process achieves its aim is referred to as design method validation. To better understand the definition of a design method, it can be described as a concept that is part of or includes other types of support like methodologies, guidelines, processes, and tools [14]. In order to achieve a high degree of validation, relevant evidence pertaining to the correctness of these design methods with the original expectations must be gathered.

Procedure: The following analytical parameters of a method are frequently evaluated during method validation: accuracy, precision, specificity, detection limit, quantitation limit, linearity, range, and robustness. Depending upon the type of method, its application, and purpose, not all the analytical characteristics indicated above will be required for validation.

Furthermore, in the field of design research, validation is divided into two categories, namely, 1) Structural validation and 2) Performance validation. Structural validation is concerned with the effectiveness of the method. This step includes the theoretical investigation of the logic and consistency of the method. Performance validation targets the efficiency of the method. This includes empirical investigations to quantify the effect of method application using the example problems. Further activities are concerned with establishing a connection to problems in practice by argumentation [14].

Challenges: The lack of a consistent approach makes it difficult to validate design methodologies in design research. There are multiple factors affecting methods validation, in particular, but not limited to, 1) the designer's cognitive and emotional interactions with the design problem, as well as his/her interactions with team members, 2) The design problem, which is typically unstructured, complex, and

one-of-a-kind may evolve during the process as a result of new insights gained through solution design or environmental factors, 3) The situation in which the design problem is addressed: From simple tasks completed in a few minutes by a single designer to large-scale design projects carried out by entire firms over several years [14]. Thus, one of the most difficult components of design method validation is attaining the appropriate level of simplification in order to establish a scientifically sound conclusion on the method's impacts without overlooking critical factors that emerge in design practice. Finally, method validation has been supported by a variety of research methodologies. However, the explanations of necessary stages and relevant research methodologies for validation varies between the approaches. Also, according to Gericke, Eckert, and Stacey (2017), there is a lack of consensus on the research methodologies and evidence required for the successful validation of design processes [14, 15].

C. Software engineering

Purpose: The FDA defines software validation as "Confirmation by examination and provision of objective evidence that software specifications conform to user needs and intended uses, and that the particular requirements implemented through software can be consistently fulfilled."

Procedure [16]: Experimentation, Data Collection, and Documentation are three key components that contribute to validation in software engineering. Furthermore, replication is important in validation; i.e. the referenced paper portrays that, if there are enough replications, the validity of the process under investigation may be determined. Aside from that, some research highlights how quantitative analysis can be used to give evidence for validation. These inconsistencies in software validation approaches emphasize the need for more stringent validation methodologies in this field. In this regard, some of the software validation challenges are listed below.

Challenges [17]: A few of the difficulties encountered during software validation are as follows. First, because the validation methods suggest a document format, reusability is a significant task to implement. The document quality alternates between the initial (true copy) and unacceptable quality when using the reusability strategy. Second, due to bias, each person's interpretation of the validated document is always different. Finally, due to the effect of new knowledge generation as the project progresses, it is difficult to make changes or propose new requirements during the project since no proper implementation methodology exists to accommodate this change.

III. BACKGROUND ON PROPOSITIONAL LOGIC

This section provides the necessary background on Propositional logic which will be used in the later sections to identify novel insights on validation in SE. In the late 19th and early 20th centuries, formal logic was developed to model reasoning with mathematically precise structures. Modern formal logics are comprised of three components: 1) a set of recursively defined sentences or symbolic representations for a set of base symbols that make up a formal language; 2) a precise and rigid semantics that gives meaning to the sentences; and 3) a proof theory that connects a set of sentences (premises) to another sentence (conclusion) [18]. In the domains they represent, formal logic can be utilized as a strong reasoning tool. It can be used to reason about domains that involve only simple propositions to scenarios that involve modalities like necessary, always will be, prefers, knows, believes, etc. Past researchers in SE and Design have used and developed formal logic systems to represent and reason about preferences [19], knowledge, and beliefs [20], which are critical elements required for decision-making, in the context of SE.

Propositional logic [21, 22] was developed primarily to reason about propositions that will otherwise be difficult or intractable when plain English is used. Some of the reasoning capabilities include: making inferences, checking for inconsistencies in premises, etc. The syntax for Propositional logic consists of a non-empty set (Φ) of atomic propositions that represent basic facts about the situation under consideration and are usually denoted by *p, q, r,* etc. "*It is raining*" and "*Mars is a planet*" are examples of atomic

propositions. Compound sentences or formulas typically represented using Greek symbols φ, ψ , etc., can be formed by closing off under conjunction and negation. The semantics for Propositional logic basically consists of a valuation function (v), that takes the propositions as its argument, and assigns a truth value. For example, for the proposition, *p*: Mars is not a planet; *v(p) = False.*

The proof theory of Propositional logic consists of several rules of inferences that are typically to reason about propositions. The following are some of the rules that will be used in Section IV.

$$\textit{Modus Ponens: } ((p \to q) \wedge p) \to q \tag{1}$$

$$\textit{Hypothetical Syllogism: } ((p \to q) \wedge (q \to r) \to (p \to r) \tag{2}$$

$$\textit{Disjunctive Syllogism: } \big((p \vee q) \wedge \neg q\big) \to q \tag{3}$$

$$\textit{Commutation: } (p \wedge q) \to (q \wedge p) \tag{4}$$

$$\textit{Simplification: } (p \wedge q) \to p \tag{5}$$

$$\textit{Addition: } p \to (p \vee q) \tag{6}$$

Definition 1 (Argument): In formal logic, an argument consists of a set of statements called premises, and conclusions, where the premises are used to demonstrate the truth of the conclusions. The example below represents an argument.

Example 1: Martian status
Premise 1: If Tom is a Martian, then he was born on Mars.
Premise 2: Tom is a Martian.
Conclusion: Tom was born on Mars.
Deductive or inductive reasoning is used to reach a conclusion. Deductive reasoning leads to a conclusion by starting with general statements and applying them to specific situations, but inductive reasoning leads to broad conclusions by starting with specific observations. Deductive reasoning will be exclusively employed for the sake of this study. Inference is the reasoning process that involves drawing conclusions from a set of premises. The logic system's axioms and rules of inferences are used to derive new conclusions from the premise. In the preceding example, modus ponens (Eq. 1) is used to derive the conclusion from the premises.

Definition 2 (Logical validity): An argument is *valid* if and only if it takes a form that makes it impossible for the premises to be true and the conclusion to be false. In the example above, the argument is *valid* since if both the premises are true, then the conclusion is true. *Validity* of arguments is defined structurally and is not concerned with the actual meaning of the sentences in the argument. Logicians typically tie logical consequence (logical entailment) to define *validity*. Specifically, there are two logical consequences: 1) Semantic or model-theoretic consequence, and 2) Syntactic or proof-theoretic consequence. With semantic consequence, an argument is *valid* if and only if there is no counter example. Mathematically, a semantic consequence can be represented as in Equation (7):

$$\Gamma \models A \tag{7}$$

Here A is a semantic consequence of a set of premises Γ. "A" is a semantic consequence if and only if for every interpretation that makes Γ true, it also makes A true. Syntactic consequence, on the other hand, can be represented by Equation (8):

$$\Gamma \vdash A. \tag{8}$$

Here A is a syntactic consequence of a set of premises Γ if and only if A can be derived from Γ by the application of inference rules that are part of the formal system. Most logicians typically associate *validity* of arguments with semantic consequence, as syntactic consequence focuses more on the formal procedure

associated with just symbol manipulation. In this paper, we will use the notion of semantic consequence to define logical validity.

Definition 3 (Soundness): An argument is *sound* if and only if it is logically valid, and its premises are true.
Lemma 1: *All invalid arguments are unsound.*
Proof: Trivial from the definition of logical validity and soundness.

Lemma 2: *Logically valid arguments with false premises are unsound.*
Proof: Trivial from the definition of logical validity and soundness.

Definition 4 (Consistency): A set of statements is consistent if and only if all the statements in the set are conjointly true.

Lemma 3: *An inconsistent set of premises always leads to unsound arguments.*
Proof: Trivial from the definition of consistency and soundness.

IV. FOUNDATIONS FOR VALIDATION IN SYSTEMS ENGINEERING

System validation is the process of checking whether all the stakeholders' needs have been satisfied. It is critical to perform validation throughout the system lifecycle, as performing only when the SOI is complete would lead to expensive iterations. There are several factors that affect validation throughout the lifecycle, the most critical artifacts being stakeholders' needs, requirements, design, and verification. To confidently claim that the SOI is valid, validity with respect to the abovementioned artifacts needs to be established first. This paper specifically focuses on these artifacts and explores the relationship between them and system validation by providing theoretical insights.

A. Stakeholders' needs

The primary goal of SE is to develop a solution that satisfies stakeholders' preferences [1, 2]. The development of any large-scale complex engineered system, such as a satellite, weapon system, etc., using SE practices starts with the elicitation of a stakeholder's (e.g.: customer, federal governing body, contractor and sub-contractor, team manager, system architect, design engineer, V&V engineer, etc.) preference statements [1, 23]. Stakeholders typically have some preconception of the outcome they desire, posited as their preferences. These preference statements represent an ordering of the needs, wants, and likes of the stakeholders over aspects of the system.

At this stage, the most critical question to be addressed with respect to validation is the following:

- What is the impact of the stakeholders' needs on the realization of a solution? In other words, can we confidently claim whether the right solution can be achieved using a given set of needs?

One of the critical aspects that is relevant to the above questions is the notion of consistency in the needs of the stakeholders. During elicitation, it is highly possible that the stakeholder's elicited needs contradict each other, especially given the textual nature of the statements. The following theorem will explore how inconsistent stakeholders' needs affect the realization of a solution.

Theorem 1: *Inconsistency in stakeholders' preferences will lead to incorrect solutions.*
Proof: This can be proved using the principle of explosion in propositional logic. Let us consider a proposition p that represents the need of stakeholder S1, and $\neg p$ represents the need of stakeholder S2. Principle of explosion states that p and $\neg p$ implies q, where q is any conclusion.

Example 2: A communication system.
p: *The system shall have uninterrupted communication*
¬*p*: *The system shall shut down for 1 hour per day*

To prove that an inconsistency in the above statements lead to any solution, i.e., incorrect solution, we will use the rules of propositional logic. The procedures for the proof are presented below, along with their justification.

1.	*p* and ¬*p*	*(Premise)*
2.	*p*	*(From 1, and simplification rule)*
3.	*p* OR *q*	*(From 2, and addition rule), where q is any proposition (e.g.: q – Mars is a planet)*
4.	¬*p*	*(From 1, commutation and simplification)*
5.	*q*	*(From 3, 4, and Disjunctive syllogism)*

Hence proved (From 5, the implied result *q* is an incorrect solution)

This proof demonstrates that when we start with inconsistent stakeholder needs, we can end up with any requirement statement (in this case, *q*) that may be irrelevant to the problem of interest, ultimately leading to a solution that is incorrect. The point of consistency in this research is not to prescribe what one should prefer, instead, it suggests that if a stakeholder has a particular set of needs (that are inconsistent), it will result in incorrect or no solutions. There are no restrictions on what a stakeholder can or should prefer.

B. Requirements

System requirements are defined and derived from the stakeholders' needs to be further decomposed into subsystem/component-level requirements. To ensure that correct requirements are defined and derived, it is critical to establish validity of the requirements at this stage. The goal is to make sure that the requirements accurately represent what is expressed by the stakeholders' needs. The following are the questions that need to be answered to ensure this:

1. Are the set of requirements traceable to the needs?
2. Can one get a solution that is not specified by the needs, even though the requirements are met?
3. Can one get a solution specified by the needs without satisfying the requirements?
4. How do inconsistencies in a set of requirements affect the satisfaction of stakeholders' needs?

In this paper, these questions will be explored using formal logic. Formal logic explicitly deals with the notion of *logical validity*, *soundness*, and *consistency*, which are notions that are directly applicable to the questions above.

<u>*Logical validity:*</u> The first notion that is relevant to Question 1 is *logical validity* (Definition 2), which ensures that it is impossible for the conclusion to be true and the premises to be false. For example, consider the following stakeholder need (N) and requirements (r1 and r2) statements.

Example 3: Speed and cost of a system
N: *Stakeholder S1 prefers maximum speed and low cost.*
r1: *The system shall have faster than light travel*
r2: *A system with faster than light travel will have zero cost*

Here the needs statement (premise) entails the requirements statements (conclusion). That is, if N is true, then r1 and r2 are true. This is the notion of logical entailment, which can ensure traceability of requirements to stakeholders' needs. It is possible to have requirements that are not directly traceable to stakeholders' needs, but through other requirements. Logical validity can be used to ensure indirect traceability as well. In the above discussion, we considered needs to be the premises, and the requirements set to be the conclusion. On the contrary, if we consider the requirements statements to be premises, and the needs to be conclusion, then we can address Questions 2 and 3. Having the requirements as premises, we can use the notion of logical validity to establish that it is impossible for the needs to be false when the requirements are true. This addresses Question 2. Although the requirement statements r1 and r2 are patently false, logical validity does not prevent from having such arguments. In other words, patently false premises do not ensure invalidity, and one can encounter worthless valid arguments even with a check for logical validity. Therefore, a stronger notion is needed to address Question 3.

Soundness: We will use soundness (Definition 3), in addition to logical validity, to establish this. The notion of soundness demands that the arguments are logically valid, and the premises are true. This will overcome the issue of having patently false premises. This ensures that one cannot have needs that are true, and the requirements that are false. In other words, one cannot satisfy the needs without satisfying the requirements. This addresses Question 3.

Consistency: Logical validity and soundness only pertain to individual statements, whereas consistency (Definition 4) is associated with a set of statements. In current practices, the process of defining and deriving requirements from stakeholders' needs is based on heuristics. Additionally, requirements are represented using textual shall statements. This makes it highly likely for the existence of inconsistencies between the requirements in a set. For example, consider the following requirement set.

Example 4: Electrical system
N: *Stakeholder S1 prefers a waterproof charging system*
r1: *The system shall be made using rubber*
r2: *The system shall work underwater*
r3: *The system shall conduct electricity*

Note: Although this example is exaggerated, the notion still holds true for the situations listed above.
One can easily see that there is a conflict between requirements r1 and r3. Although rubber can be used to build a waterproof system, rubber being an insulator does not help conduct electricity. Thus, the need cannot be recognized due to the inconsistencies found in the set of requirements. Similar to inconsistencies in stakeholders' needs, having inconsistencies in a set of requirements is not desirable. The following theorem emphasizes this point further.

Theorem 2: *Inconsistency in a set of requirements may lead to the satisfaction of inapplicable stakeholders' needs.*
Proof: Since this proof is very similar to Theorem 1, which uses the principle of explosion, we will leave this to the reader. Here inapplicable stakeholders' needs can be any needs statements that are not relevant to the problem of interest.

Therefore, it is crucial to have a consistent set of requirements to facilitate satisfaction of applicable stakeholders' needs. Question 4 is addressed using this notion of consistency.

Definition 5 (Requirements Validity): For a set of requirements (R) to be valid, it needs to be traceable to the stakeholders' needs (logically valid), sound, and consistent. Mathematically, a set of requirements (R) is valid, if and only if

- $\exists\, n_x \in N, s.t. n_x \vDash R$
- $R \vDash n_x$
- $R\ is\ true$
- The set R is consistent

Where N is the set of all stakeholders' needs, and R is the set of requirements

C. Design and Verification

Designs are defined in terms of the design variables, as vectors. For example, D = [Wing type, Wing material, chord, wingspan] defines the design of an airplane wing, where the elements Wing type, Wing material, etc., are the design variables with a range for each of them. For example, there may be two choices available for a wing – rectangular, and swept wing. The process of design involves identifying the optimal combination of design variables that satisfy the requirements. At this stage, with respect to validation, one needs to ensure that the right designs are achieved, i.e., the designs that are intended by the stakeholders' needs.

- Can one achieve the right design by satisfying the specified set of requirements?

The only way the above question can be answered is using the notions of *logical validity, soundness,* and *consistency.* Before answering the question of right design, first we will focus on checking the satisfaction of requirements. The process of checking whether the requirements are satisfied is called *verification*. It involves performing one or more activities which may be in the form of inspection, analysis, modeling and simulation, tests, etc., to generate certain observable results that provide insights into the status of the actual requirement. To generalize, a *verification activity* is performed, and an observable *verification evidence* is generated that leads to the satisfaction or dissatisfaction of a *verification criteria*, which in turn acts as a proxy to conclude whether the requirement is satisfied or not. Here the requirement is not directly observable. Let us consider the following example of a weighing scale system:

Example 5: Weighing scale system

N: *Stakeholder S1 prefers a system capable of weighing a maximum of 260 lbs*
r1: *The structure of the system shall withstand a mass of 300 lbs*

In the above example, r1 details a mass of 300 lbs as opposed to the initial 260 lbs for additional tolerance. It is noted that the requirement r1 is not directly observed. Thus, to perform verification activities, an initial design model of the weighing scale is created. Following this, a verification activity is performed to check whether the model withstands such a load.
Verification activity: *Perform a finite element analysis on the computational model*
Verification criterion: *The deformation shall be less than 0.1 mm*
Verification evidence: *The deformation is 0.3 mm*
If the results from the analysis (verification evidence) shows deformation, then this leads to dissatisfaction of a verification criteria. The verification activities can be performed again by changing the design parameters and material of the model to achieve a higher degree of satisfaction. Given all these elements of verification, a right design can only be achieved if and only if the verification criteria are valid. The procedure to Establish the validity of verification criteria is similar to requirements validity.

Definition 6 (Valid verification criteria): A set of verification criterion is valid if and only if it is traceable to at least one requirement, entails the set of all requirements, is sound, and consistent. Mathematically, a set of verification criteria (VC) is valid if and only if

- $\exists\, r_x \in R, s.t. r_x \vDash VC$
- $VC \vDash R$
- $VC\ is\ true$
- The set *VC* is consistent

Definition 7 (Design Validity): A design, D, is valid if and only if it satisfies the requirements that are specified by the stakeholders' needs. Mathematically, a design (D) is valid if and only if

- $D \vDash R$
- $D\ is\ true$
- D is consistent

Theorem 3: *A design is valid if and only if (a) it logically entails the verification criteria, and (b) the set of verification criteria is valid.*

Proof: This can be proved using the rules of propositional logic followed by Lemma 1, 2, and 3 detailed in Section III. The procedures for the proof are presented below, along with their justification.

1.	$D \vDash VC$	*(Premise: from condition (a) of theorem 3*
2.	$VC \vDash R$	*(Premise: from condition (b) and definition 6: Valid verification criteria)*
3.	$VC\ is\ true\ and\ consistent$	*(Premise: from condition (b) and Definition 6: Valid verification criteria)*
4.	$D \vDash R$	*(From 1,2, and Hypothetical Syllogism*
5.	$R\ is\ true$	*(From 2 and 3)*
6.	$D\ is\ true\ and\ consistent$	*(From 4, 5 and Lemma 1,2 and 3)*

Hence proved (From 4 and 6, the implied result conforms with definition 7: Design Validity).
This ensures that the right design is achieved, i.e., the design that satisfies the valid verification criteria that act as proxies for the valid requirements that accurately represent the stakeholders' needs.

D. System Validation

System validation is the process of checking whether the final SOI has met all the stakeholders' needs. Similar to verification, *validation activities* (e.g.: demonstration, test, etc.) are performed to check for the satisfaction of *validation criteria*, which are proxies for the stakeholder needs statements. Based on the *validation evidence*, the stakeholders and contractors agree to either proceed with deployment or reiterate. Current practices are heuristics-based and rely heavily on human judgment, which may lead to suboptimal solutions. Therefore, it is critical to identify optimal validation procedures to ensure that the right system is built, without incurring cost and time overruns.

Definition 8 (System validity): A system is valid if and only if it meets all the stakeholders' needs. Mathematically, a system is valid if and only if

- $S \vDash N$
- $S\ is\ true$
- S is consistent

Definition 9 (Valid validation criteria): A set of validation criterion is valid if and only if it is traceable to at least one need, entails the set of all needs, is sound, and consistent. Mathematically, a set of validation criteria (VDC) is valid if and only if

- $\exists\, n_x \in N, s.t. n_x \vDash VDC$
- $VDC \vDash N$
- $VDC\ is\ true$
- The set *VDC* is consistent

Theorem 4*: A system is valid if and only if (a) the stakeholders' needs are consistent; (b) requirements are valid; (c) design is valid; and (d) the set of validation criteria is valid, and (e) the system entails validation criteria.*

Proof: This can be proved using the rules of propositional logic followed by Lemma 1, 2, and 3 detailed in Section III. The procedures for the proof are presented below, along with their justification.

1. Requirements are valid	*Premise*
a. $R \vDash N$	
b. $R\ is\ true$	*(Definition 5: Requirement Validity)*
c. $R\ is\ consistent$	
2. Design is valid	
a. $D \vDash R$	*Premise*
b. $D \vDash VC$	*Theorem 3*
c. $VC \vDash R$	
d. $D, VC, R\ are\ true$	
e. $VC\ is\ consistent$	
3. Validation criteria is valid	
a. $VDC \vDash N$	*Premise*
b. $VDC\ is\ true$	*Definition 9: Validation criteria)*
c. $VDC\ is\ consistent$	
4. $S \vDash VDC$	*(From condition (e) of theorem 4)*
5. $N\ is\ true$	*(Needs are consistent, and from equation 1b)*
6. $S \vDash N$	*(From 4, 3a, and Hypothetical Syllogism)*
7. $S\ is\ true\ and\ consistent$	*(From 5, 6 and Lemma 1,2 and 3)*

Hence proved (From 6 and 7, the implied result conforms with Definition 8: System validity)

Definition 10 (Verified system): A system is completely verified if all the requirements are verified to be true. Mathematically, a system is verified if and only if

- $R\ is\ true\ and\ consistent$
- VC is true and consistent

Theorem 5: *A valid system implies a completely verified system*
Proof: This can be proved using the rules of propositional logic followed by Lemma 1, 2, and 3 detailed in Section III. The procedures for the proof are presented below, along with their justification.

1. Requirements are valid	*(Theorem 4: since a valid system is given)*
a. $R \vDash N$	*(Definition 5: Requirement Validity)*
b. $R\ is\ true$	
c. $R\ is\ consistent$	
2. Design is valid	*(Theorem 4: since a valid system is given)*

a. $D \vDash R$ *(Definition 7: Design Validity)*
b. $D \vDash VC$
c. $VC \vDash R$ *(Theorem 3)*
d. D, VC, R *are true*
e. VC *is consistent*

3. R *is true and consistent* *(From 1b, and 1c)*
4. VC *is true and consistent* *(From 2d, and 2e)*

Hence proved (From 3 and 4, the implied result conforms with Definition 10: Verified system)

Corollary 1: *A completely verified system does not imply a valid system*

Proof: For the below argument to be valid, the conclusion (valid system) must always be true when the premises (verified system) are true. Given the definitions and the previous theorems, one can easily construct a truth table and verify that this is an invalid argument.

$$Verified\ System \nvDash Valid\ System$$

As previously noted, the foundation for validation in Systems Engineering is substantially insufficient. This section attempts to address the lack of rigorous methodologies and the requirement for a mathematically established foundation for validation. With the help of the four artifacts, namely stakeholder needs, requirements, design, and verification, system validation has been successfully defined. Furthermore, rigorous set of theorems and a logically proven theoretical framework exploring the relationship between artifacts and overall system validation is provided. The idea behind what validation means is mathematically described using properties such as logical validity, soundness, and consistency using Propositional logic. Following that, the definition of validation in each of the above-mentioned artifacts is mathematically demonstrated. For each artifact, a collection of theorems is proposed, which serve as the foundational evidence for the pre-defined ideas of validation. The reasons for defining validation as such are better described by a collection of descriptive examples provided for each artifact. Finally, the overall definition of what system validation entails in SE is explicitly articulated, based on the collection of theorems proven previously. Table IV.1 summarizes the theoretical insights.

Table IV.1: Theoretical insights for validation in SE

Theoretical insights	Description
How does inconsistency in stakeholders' needs, affect the realization of solutions?	***Theorem 1:*** *Inconsistency in stakeholders' preferences will lead to incorrect solutions*
Requirements traceability	***Logical validity*** (Definition 2)
Can one get a solution that is not specified by the needs, even though the requirements are met?	***Logical validity*** (Definition 2)
Can one get a solution specified by the needs without satisfying the requirements?	***Soundness*** (Definition 3)
How do inconsistencies in a set of requirements affect the satisfaction of stakeholders' needs?	***Theorem 2:*** *Inconsistency in a set of requirements may lead to the satisfaction of inapplicable stakeholders' needs.*
When is a design valid?	***Theorem 3:*** *A design is valid if and only if (a) it logically entails the verification criteria, and (b) the set of verification criteria is valid;*
When is a system valid?	***Theorem 4****: A system is valid if and only if (a) the stakeholders' needs are consistent; (b) requirements are valid;*

	(c) design is valid; (d) the set of validation criteria is valid; and (e) the system entails the validation criteria.
Relationship between verification and validation	***Theorem 5:*** *A valid system implies a completely verified system* ***Corollary 1:*** *A completely verified system does not imply a valid system*

V. THE PROBLEM OF VALIDATION – TEXTUAL STAKEHOLDERS' NEEDS, AND PREFERENCE FUNCTIONS

Since system validation ultimately depends on the satisfaction of stakeholders' needs, the way in which these needs are represented plays a huge role in whole process. In current SE practices, stakeholders' needs are represented using textual statements that are maintained in documents. Recently, researchers have proposed value-based approaches that use preference/value functions to mathematically capture the stakeholders' preferences. This section will focus on discussing the challenges associated with both these approaches, followed Section IV that discusses how the recently developed *Modal preference logic* [19] can overcome some of the challenges in validation.

A. Textual stakeholders' needs

In current practice, the desires of stakeholders are represented using textual *stakeholder needs* statements. Stakeholder needs are further defined and derived into system requirements, which are then decomposed into subsystem and component level requirements that flow down the organizational hierarchy to aid in decision-making[23, 24]. The system of interest is realized only when all stakeholder needs are satisfied, which, in turn, depends on satisfying lower-level requirements. The textual nature of needs statements has the following limitations: easier to misinterpret, lack of a rigorous mechanism to facilitate updates in the needs, difficulty in communicating needs to other decision-makers, etc. Additionally, the textual nature of needs statements results in a lack of analysis capabilities including, but not limited to, evaluation of inconsistencies in stakeholders' preferences, identification of optimal solutions that satisfy stakeholders' preferences, changes/updates in preferences, decision traceability, etc. In addition, as discussed in Section IV (specifically Theorem 4), system validation depends on the validity of stakeholders' needs, requirements, design, and verification. Ensuring all these validities is challenging while using textual statements.

In essence, it is extremely difficult, if not impossible, to confidently and accurately answer the following questions using textual needs statements: "*Have we built the right system?;* and "*What is the optimal validation strategy?".*

B. Preference/value functions

Recently, researchers have proposed a value-based alternative[25-31], to requirement-driven approaches in SE, wherein the elicited stakeholder preference statements are translated to a mathematical preference function (a special objective function), which is then used for optimization. Preference functions are created by assessing the tradeoffs between the key attributes from the decision-maker in an *ad-hoc* manner. A preference function allows for direct comparison of design alternatives from a wide range of systems that share the same set of system characteristics. It should be noted that the output of a preference function is a scalar value that does not have an inherent meaning but provides a rank ordering of prospects. On the other hand, a value function is a special case of a preference function, where the unit of measurement is in monetary terms. For example, Profit, Net Present Profit, etc., are used as value functions. The challenges discussed below suit both preference and value functions.

Definition 10 (Preference function): A preference function, V(X), is a special case of an objective function that captures the tradeoffs specified by the decision-maker. Here, the argument X represents the set of attributes that characterize the problem of interest.

The following table discusses the challenges associated with using preference functions in large-scale SE projects.

Table V.1: Preference function validity - Challenges

Challenges	Consequences
Multi-stakeholder preference aggregation: SE projects involve multiple stakeholders, where the preferences of multiple stakeholders need to be elicited and somehow aggregated to form a holistic preference function. This is extremely challenging given that there is a lack of a mathematically rigorous procedure that does not violate rationality conditions [32, 33].	Value-based approaches do not scale well to multi-stakeholder scenarios. There is still a lack of validation procedures to confidently claim the validity of aggregated preference functions.
Formulating preference functions – single decision-maker: The process of creating preference/value functions starts with eliciting the preference statements from the decision-maker, where preferences are expressed only over outcomes, and always have a direction specified – e.g., Stakeholder A prefers high profit. This is followed by the identification of key attributes (e.g.: Cost, quality, etc.) that are of concern to the decision-maker. Preference/value functions are then created by assessing the tradeoffs between these key attributes from the decision-maker. Although the idea of using a preference function is very appealing, given its foundations in Decision Analysis, the whole process of creating a preference function is ad-hoc in nature lacking a well-grounded generic framework.	This leads to the possibility of multiple preference functions that may be relevant to the decision problem of interest, which ultimately makes it difficult to decide on which is the right one. Some preferences might be overlooked and subjected to trade-offs that are entirely subjective. Hence, without determining the validity of preference functions, selecting the right system using preference functions is meaningless.
Attributes and their interactions: In large-scale complex engineered systems, the preference function is a function of many attributes, leading to a lengthy and messy preference function. Furthermore, it is challenging to understand the relationship between all these different attributes. The proxy attributes and models provided by the designers and other entities to formulate the preference function may not necessarily map the needs given by the stakeholders.	Such a preference function would be more difficult to comprehend, making it impossible to determine what the preference function intended to represent. Furthermore, it is difficult, if not impossible, to answer the question: *Have we captured all the attributes, and their interactions correctly and accurately?*
Meaning of preference function: Can a final number clearly explain the end goal of a system? For example, imagine a function that addresses the right car to be bought from a set of cars A, B, and C. If the preference function gives a high value for car A compared to car B and C, does knowing that number accurately describe how one will feel or experience if car A is bought in a real-world scenario? After all, a preference value is just a representation of what shall be following a set of hypothetical scenarios and not what is in a real-world case.	Thus, by not knowing how exactly it will pan out, the preference function cannot be blindly trusted and the decision-maker has no way of distinguishing or comparing the attributes [19]. Hence, some form of validation procedure is required that better explain that the right system is built.

Thus, a preference function only provides the rank ordering of the prospects and not how much one prospect weighs when compared to another. A value function overcomes this challenge, however mapping everything to monetary terms is still questionable and remain challenging, especially when the missions are not primary profit-driven – Space exploration, Search and rescue, etc.	
Stakeholder preference communication: Using preference functions to communicate stakeholders' preferences to other entities (designers, managers, V&V engineers, etc.) in an organization has significant pragmatic issues, given the following challenges – multi-dimensional decisions, mathematically relating all the decisions to the value function through physics, organizational and federal policies, capturing tradeoffs, testing the accuracy and validation of the value functions, and ensuring internal physics-based consistency in the overall system optimization.	

Before addressing the question of validation of an SOI using preference functions, it is critical to first define validity of a preference function.

Definition 11 (Valid preference function): Given a finite number of prospects, a preference function is valid if and only if it rank-orders prospects exactly as the decision-maker does. Proving this is extremely difficult given the challenges highlighted in Table V.1. In addition, since it is a function that models the preferences of the decision-maker, it is not truth preserving. That is, one cannot establish the truth value of the model, thus making it impossible to claim *logical validity* (Definition 1). One can use the validation techniques from the domain of Modeling and Simulation (Section II), however the same challenges that were discussed in that section still exist.

Given the discussions in Table V.1, one can confidently claim that establishing the validity of preference is currently not possible. A system is validated if and only if it is realized through the optimization of a valid preference function. Since ensuring validity of a preference function is not possible, performing system validation is meaningless when using preference functions.

Another school of thought is that verification and validation activities can be used as knowledge gathering activities that provide evidence to update the probability distributions on attributes, and the overall value respectively. This is appealing, however without establishing the validity of preference/value functions, it is still not as useful.

VI. FUTURE DIRECTIONS - MODAL PREFERENCE LOGIC AS A MECHANISM FOR VALIDATION IN SE

Recently, an approach based on formal logic was proposed to represent stakeholders' preferences in SE, called *Model preference logic* [19]. This was achieved by extending the existing logics of preference (specifically semantic approaches) in Philosophy that is based on the classical possible-worlds model [34-36]. This approach has improved expressive power over the existing requirements-based and value-based approaches, by facilitating representation of a wide range preference statements as seen in Table VI.1.

Table V.1: Preferences in Systems Engineering [19]

Type of Preferences		Example (Stakeholder X)
Absolute	Unconditional	Target-oriented: X *prefers* uninterrupted communication; Design-dependent: X *prefers* Solar arrays for power generation; Objective-oriented: X *prefers* low total satellite mass;
	Conditional	Target-oriented: If the satellite is parked in LEO, then X *prefers* uninterrupted communication; Design-dependent: If transponder 'y' is used, then X *prefers* solar arrays for power generation; Objective-oriented: If the satellite weighs more than 1000 kg, then X *prefers* high signal quality;
Comparative	Unconditional	Target-oriented: X *prefers* a system mass less than 1000 kg to uninterrupted communications; Design-dependent: X *prefers* Solar arrays to Nuclear reactor; Objective-oriented: X *prefers* low total cost to high signal quality;
	Conditional	Target-oriented: If it is a multi-satellite system, X *prefers* uninterrupted communications to a system mass less than 1000 kg; Design-dependent: If it is a multi-satellite system, then X *prefers* solar arrays over nuclear reactors; Objective-oriented: If the satellite weighs more than 1000 kg, then X *prefers* high signal quality to low total cost;

The *modal preference language* (L_P) is given by the following Backus–Naur/Backus–Normal Form (BNF) [37].

$$\varphi := p \mid \neg\varphi \mid (\varphi \wedge \psi) \mid Pref(\varphi) \tag{9}$$

BNF is a formal notation representing the grammar of the formal language. := means "may be expanded to" or "replaced with". In the above notation, the formula φ can be replaced with simple propositions p, and/or compounded formulas with the preference operator. The semantics for the *modal preference language (Lp)* was developed using the classical *possible worlds* approach [34-36]. For instance, *comparative preference* in Table 1 is defined using Eq. 10.

$$(M, w) \vDash \psi[Pref]\varphi \Leftrightarrow \forall w' \& w'': (M, w') \vDash \varphi \,\&\, (M, w'') \vDash \psi, \quad it\ is\ the\ case\ that\ w' \succcurlyeq w'' \tag{10}$$

Equation 1 means that "An agent prefers φ to ψ if and only if all the states where φ holds is at least as good as all the states where ψ holds". In other words, all φ-states are as good or better than all ψ-states. In Eq. 1, M represents a *preference structure*, which is a tuple $M = (S, \succcurlyeq, \pi)$, where S is the set of all states ($S = \{w_1, w_2, w_3, \ldots w_n\}$), also sometimes called the domain of M. In the context of large-scale systems, S can be considered as the set of all alternatives that a decision-maker considers possible, i.e., S represents the entire solution space. Here $\succcurlyeq$ is a binary relation that has a total order, called the *betterness relation* that is used to evaluate preferences as defined by Eq. 11.

$$\succcurlyeq (w) = \{w' : (w, w') \in \succcurlyeq\} \quad \succcurlyeq \subset S \times S \tag{11}$$

Here, $w' \succcurlyeq w$ is read as w' is *at least as good* as w. If $w' \succcurlyeq w$ and $w \not\succcurlyeq w'$, then w' is *strictly better* than w, i.e., $(w' \succ w)$. Since the binary relation $\succcurlyeq$ has a total order, it is *reflexive, transitive, antisymmetric,* and *total*. The preference structure *M* contains all the necessary elements required to represent preference as a *modal* and is used to evaluate the preference statements elicited from the stakeholder. The π in the structure *M* is a valuation function, that assigns truth values to each of the atomic propositions in Φ (the set of all atomic propositions) at each state, i.e., $\pi(w, p) = TRUE$ means that p is true at state w.

The following points emphasize why *Modal preference logic* is an appropriate tool to achieve a normative approach to validation in SE:

- Given the formal structure of Modal preference logic, and demonstrated by past work [19], one can explicitly represent (i) a wide variety of stakeholders' preferences (Table VI.1) using sentences involving the preference operator; (ii) requirements using simple and compound statements; and (iii) design using possible worlds and propositions.
- Based on the theoretical insights in Section IV, a normative approach to validation must facilitate the notions of logical validity, soundness, consistency, and must be truth preserving. Based on the semantics of *Modal preference logic*, all of these can be ensured.
- The logic system discussed in the previous section enables reasoning capabilities in both syntactic and semantic terms. Reasoning in syntactic terms is done by deriving conclusions from premises by applying the axioms and rules of inferences of the logic system. In semantic terms, there are two main reasoning capabilities – (1) Checking for satisfiability; and (2) Model checking [35].
    - *Checking for satisfiability:*
      In satisfiability problems, one will reason through the question - "*Is a formula φ satisfiable*" or "*Are the set of formulas Φ satisfiable*? A formula φ is *satisfiable* if there exists some structure M and some state $w \in S$ for M such that $(M, w) \vDash \varphi$ [22, 35, 38, 39]. A set of formulas Φ is *satisfiable* if and only if there exists some structure M and some state $w \in S$ for M such that $\forall_{\phi \in \Phi}(M, w) \vDash \phi$ [22, 35, 38, 39]. Given a set of sentences, one can determine whether the set is consistent or not using this notion of satisfiability. If the set of sentences is inconsistent, then the set of sentences are not satisfiable. This notion of satisfiability can be used to determine inconsistencies in stakeholders' needs, requirements, verification, and validation criteria.
    - *Model checking:*
      In model checking problems, one reasons through the question – *"Is a sentence φ true in a pointed model* $(M, w) \vDash \varphi$*?"*. This can be expanded to identify the set of worlds where a formula φ is true. Past work has demonstrated how one can use Modal preference logic to identify designs that satisfy stakeholders' preferences. This can be expanded to identify optimal validation strategies.

Table VI.1: Modal preference logic as a mechanism for validation in SE

	Textual statements	**Preference/value functions**	**Modal preference logic**
Expressive power to represent a wide variety of stakeholders' preferences, requirements, etc.	Plain English, not rigorous	Limited	✓
Logical validity	-	-	✓
Soundness	-	-	✓
Consistency	-	-	✓
Ensure validity mathematically	-	-	✓
Enable methodologies to identify optimal validation strategies	-	Limited	✓

VII. CONCLUSION

This paper has focused on characterizing validation in Systems Engineering in terms of SE artifacts including stakeholders' needs, requirements, design, and verification. Formal definitions for validities associated with the artifacts, and the overall system were provided using Propositional logic as the foundation. The logical notions of logical validity, soundness, and consistency were used to define the different validities.

Some theoretical insights were derived using mathematical proofs that provide an understanding of the following questions: *What does it mean for the design to be valid? What does it mean for the system to be*

valid? How are verification and validation related? What is the impact of inconsistencies in the stakeholders' needs, and requirements on the realization of solutions?

In addition, this paper has extensively discussed the challenges associated with using textual needs statements, and preference functions as mechanisms to represent stakeholders' needs in terms of the foundations of validation developed. Specific challenges include a lack of flexible expressive power and rigor, difficulty in establishing validity, and the lack of mathematical procedures to identify optimal validation strategies. Finally, this paper emphasizes the appropriateness of using *Modal preference logic*, a previously developed formal logic system to represent and reason about stakeholders' preferences in SE, as a mechanism to achieve a normative approach to validation.

Given the foundations in this paper, future work will focus on developing methodologies to identify and resolve inconsistencies, ensure validities in all the artifacts, and optimize validation strategies by exploiting Modal preference logic.

REFERENCES

1. Haskins, C., et al. *Systems engineering handbook*. in *INCOSE*. 2006.
2. SEBok. *System Validation*. 2021 [cited 2021 Oct 19]; Available from: https://www.sebokwiki.org/wiki/System_Validation.
3. Simpson, T.W. and J.R. Martins, *Multidisciplinary design optimization for complex engineered systems: report from a national science foundation workshop.* Journal of Mechanical Design, 2011. **133**: p. 101002.
4. Paul, D.C., *Report on the Science of Systems Engineering Workshop*, in *53rd AIAA Aerospace Sciences Meeting*. 2015, American Institute of Aeronautics and Astronautics.
5. Bloebaum, C.L., P. Collopy, and G.A. Hazelrigg, *NSF/NASA Workshop on the Design of Large-Scale Complex Engineered Systems - From Research to Product Realization*, in *14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. 2012: Indianapolis, Indiana.
6. DARPA/NSF, *DARPA/NSF Systems Engineering and Design of Complex Aerospace Systems Workshop*. 2009: Arlington, VA.
7. Collopy, P., *Final Report: National Science Foundation Workshop on the Design of Large Scale Complex Systems University of Alabama in Huntsville.* Center for System Studies, 2011.
8. NASA. *Systems Engineering Postulates, Principles, Hypotheses*. 2018 [cited 2019; Available from: https://www.nasa.gov/consortium/postulates-principles-hypotheses.
9. (DoD), D.o.D., *DoD Modeling and Simulation (M&S) Verification, Validation, and Accreditation (VV&A) - Instruction*. 2009.
10. Sargent, R.G., *Verification and validation of simulation models.* Journal of simulation, 2013. **7**(1): p. 12-24.
11. Law, A.M. *How to build valid and credible simulation models*. in *2019 Winter Simulation Conference (WSC)*. 2019. IEEE.
12. Pace, D.K., *Modeling and simulation verification and validation challenges.* Johns Hopkins APL technical digest, 2004. **25**(2): p. 163-172.
13. Petty, M.D., *Modeling and validation challenges for Complex Systems*, in *Engineering Emergence*. 2018, CRC Press. p. 199-216.
14. Eisenmann, M., et al., *Design method validation–an investigation of the current practice in design research.* Journal of Engineering Design, 2021. **32**(11): p. 621-645.
15. Gerrike, K., C. Eckert, and M. Stacey, *What do we need to say about a design method?* 2017.
16. Zelkowitz, M.V. and D. Wallace, *Experimental validation in software engineering.* Information and Software Technology, 1997. **39**(11): p. 735-743.
17. Feldt, R., et al. *Challenges with software verification and validation activities in the space industry*. in *2010 third international conference on software testing, verification and validation*. 2010. IEEE.
18. Van Dalen, D., *Logic and structure*. 2004: Springer.
19. Kannan, H., et al., *Theoretical Foundations for Preference Representation in Systems Engineering.* Systems, 2019. **7**(4): p. 55.

20. Kannan, H., *Formal reasoning of knowledge in systems engineering through epistemic modal logic.* Systems Engineering, 2021. **24**(1): p. 3-16.
21. Hinman, P.G., *Fundamentals of mathematical logic*. 2018: AK Peters/CRC Press.
22. Gensler, H.J., *Introduction to logic*. 2012: Routledge.
23. Buede, D.M., *The Engineering Design of Systems : Models and Methods*. Vol. 55. 2009: John Wiley & Sons.
24. NASA, *NASA Systems Engineering Handbook*. Vol. NASA/SP-2007-6105 Rev1. 2007, Washington, D.C.
25. Kannan, H., B.L. Mesmer, and C.L. Bloebaum, *Increased System Consistency through Incorporation of Coupling in Value-Based Systems Engineering.* Systems Engineering, 2017. **20**(1): p. 21-44.
26. Murugaiyan, S., et al. *A comprehensive study on modeling requirements into value formulation in a satellite system application*. in *The 14th Annual Conference on Systems Engineering Research (CSER 2016). Huntsville, Alabama, USA*. 2016.
27. Goetzke, E.D., *Value-Driven Design of Non-Commercial Systems through Bargain Modeling*, in *Aerospace Engineering*. 2015, Iowa State University.
28. Maddox, I.D., P.D. Collopy, and P.A. Farrington, *Value-Based Assessment of DoD Acquisition Programs.* forthcoming in Procedia Computer Science, 2013.
29. Collopy, P.D. and P.M. Hollingsworth, *Value-driven design.* Journal of Aircraft, 2011. **48**(3): p. 749-759.
30. Julie, C., et al., *Application of Value-Driven Design to Commercial Aero-Engine Systems*, in *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. 2010, American Institute of Aeronautics and Astronautics.
31. Collopy, P.D., *Aerospace system value models: A survey and observations.* AIAA Paper, 2009. **6560**: p. 2009.
32. Hazelrigg, G.A., *The implications of Arrow's impossibility theorem on approaches to optimal engineering design.* 1996.
33. Black, D., *On Arrow's impossibility theorem.* The Journal of Law and Economics, 1969. **12**(2): p. 227-248.
34. Divers, J., *Possible worlds*. 2006: Routledge.
35. Ditmarsch, H., et al., *Handbook of epistemic logic*. 2015: College Publications.
36. Fagin, R., et al., *Reasoning about knowledge*. 2004: MIT press.
37. McCracken, D.D. and E.D. Reilly, *Backus-naur form (bnf).* 2003.
38. Smullyan, R.R., *First-order logic*. Vol. 43. 2012: Springer Science & Business Media.
39. Blackburn, P., J.F. van Benthem, and F. Wolter, *Handbook of modal logic*. Vol. 3. 2006: Elsevier.