DISSERTATION

COMPARATIVE ANALYSIS OF MODEL-BASED SYSTEMS ENGINEERING AND TRADITIONAL SYSTEMS ENGINEERING APPROACHES FOR ARCHITECTING ROBOTIC SPACE SYSTEMS THROUGH KNOWLEDGE CATEGORIZATION, AUTOMATIC INFORMATION TRANSFER, AND AUTOMATIC KNOWLEDGE PROCESSING MEASURES

Submitted by

Paulo Younse

Department of Systems Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Fall 2021

Doctoral Committee:

Advisor: Thomas Bradley

John Borky
Ron Sega
Steven Reising

ABSTRACT

COMPARATIVE ANALYSIS OF MBSE AND TRADITIONAL SYSTEMS ENGINEERING APPROACHES FOR ARCHITECTING ROBOTIC SPACE SYSTEMS THROUGH KNOWLEDGE CATEGORIZATION, AUTOMATIC INFORMATION TRANSFER, AND AUTOMATIC KNOWLEDGE PROCESSING MEASURES

Robotic space systems have enabled us to explore the far reaches of our solar system. However, these missions are high-cost, high-risk, and prone to accidents due to their complex nature. As these systems continue to grow even more capable and complex, spacecraft costs and mission success risk are also expected to grow. Current systems engineering approaches are finding it challenging to manage this growth in system complexity. Model-Based Systems Engineering (MBSE) offers techniques to aid in the development of complex systems, aiming to reduce design errors, reduce cost through prevention of costly rework, and improve system quality and project performance over traditional systems engineering techniques. Robotic space systems have much to benefit from an MBSE approach due to their intrinsic complexity, particularly if MBSE is implemented during the early architecting phase of the project. Case studies from the literature assert that there are benefits to using MBSE when applied to developing complex systems. However, none of these studies perform in-depth quantitative comparative analysis of applying MBSE vs. non-MBSE approaches, and there currently is a lack of substantial and compelling evidence to establish broad adoption of MBSE within the systems engineering community.

This research measures the benefits of MBSE approaches over traditional, non-MBSE approaches for architecting robotic space systems though comparative analysis, focusing on quantitative evidence supporting how MBSE better describes, develops, and evaluates the system architecture, all which can aid in the adoption of MBSE within the robotics space systems domain. These advantages will be investigated through studying 1) how an MBSE approach better captures the information content for describing a robotic space system architecture relative to a non-MBSE approach, 2) how an MBSE approach reduces the implementation effort required to developing a robotic space system architecture relative to a non-MBSE approach, and 3) how an MBSE approach more efficiently evaluates a robotic space system architecture relative to a non-MBSE approach.

A Mars orbiting sample Capture and Orient Module (COM) system for a Capture, Contain, and Return System (CCRS) payload concept for the notional Mars Sample Return (MSR) campaign develop at the NASA Jet Propulsion Laboratory was used as a case study to investigate the advantages of MBSE. The MBSE approach provided measurable advantages to architecting the COM robotic space system in terms of a higher fraction of formally captured architecture content in the appropriate knowledge category, a higher quantity of automatic information transfer between architecting tasks, and a higher quantity of automatic knowledge processing during modeling and simulation activities.

ACKNOWLEDGMENTS

The research described in this publication was carried out at the Jet Propulsion Laboratory of California Institute of Technology under contract from the National Aeronautics and Space Administration (NASA). The decision to implement Mars Sample Return will not be finalized until NASA's completion of the National Environmental Policy Act (NEPA) process. This document is being made available for information purposes only.

I would like to express my sincere gratitude to my advisor, Dr. Thomas Bradley, for his guidance and assistance through every stage of the program and research. I would also like to thank my committee members, Dr. John Borky, Dr. Ron Sega, and Dr. Steven Reising for their professional guidance and feedback. Additional thanks to Dr. Richard Mayer for guiding me through the Revised Bloom's Taxonomy and cognitive psychological principles, Jessica Cameron for the modeling and simulation support, and our Capture and Orient Module engineering team and peer reviewers for supporting our various architecting process activities.

Finally, I would like to thank my wife for her unwavering support and encouragement throughout this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

On August 5, 2012, NASA's 900 kg Mars Science Laboratory (MSL) Curiosity rover successfully landed on the surface of Mars and set out to search for evidence of past habitable environments [1] [2]. The Curiosity rover pushed the boundaries of technology and systems engineering, consisting of approximately 50,000 parts, involving nearly 3,000 NASA employees and 4,000 non-government workers, and was considered the most complex rover of its time ever sent to another planet [1] [3] [4].

Despite the technical and scientific achievements of the rover, the project experienced numerous development challenges, and in the end, saw an increase in over $881 million in costs from its original 2008 project baseline, as well as a 26-month launch delay due to design and technical problems that necessitated late design changes in hardware, avionics, and software [5]. A metric for design changes used by NASA is "drawing growth" after the Critical Design Review (CDR), with which MSL saw a 147% growth [6]. Some of these late design changes were attributed to the discovery of divergent requirements discovered late during the testing phase. These divergent requirements were found to be a consequence of not having an architecture to pull together and manage in a cohesive manner the complex web of documentation of system and subsystem functional requirements, environmental requirements, interface control documents, institutional policy documents, and planetary protection requirements [7].

On February 18, 2021, NASA landed the Mars 2020 Perseverance rover, with the goal to search for evidence of past life on Mars, as well as collect a set of samples for potential return to Earth [8] [9]. Mars 2020 reuses roughly 85% of the engineering design of MSL, but carries new

hardware and instruments for sample collection, and to search for biosignatures [10] [11]. Mars 2020 faced similar technical and design challenges as MSL did, resulting in late design changes, drawing growth related to the Sampling and Caching Subsystem and mechanical subsystems, and a $356 million cost overrun relative to its original baseline development cost estimate [12] [13]. For both of these robotic space system project examples, late design changes were a major factor in cost growth and schedule delays. In fact, NASA has observed that major projects that have rebaselined cost and schedule tended to have experienced more of these late design changes [14].

In general, space missions are considered high-risk systems and more prone to accidents due to their tightly coupled systems and need to manage complex interactions [15]. Due to the multidisciplinary nature of space missions, they exhibit complexity in requirements, design, flight software development, testing, and operations, which have been correlated to higher spacecraft cost and lower rates of mission success [15]. Robotic space systems, in particular, are becoming increasingly more complex, and hardware capability and software complexity are expected to continue to grow [16]. Using lines of code as one indicator of complexity, history shows an exponential growth trend in flight software complexity for robotic missions [15]. If this trend in complexity growth continues, higher spacecraft costs and mission success risk for future robotic space missions can also be expected to grow.

NASA's Jet Propulsion Laboratory (JPL) stated in its 2018 Strategic Implementation Plan that it will "pursue our long-term scientific Quests with a diverse and bold portfolio of missions as we push the limits of space exploration technology by developing and fielding ever more capable autonomous robotic systems" [17]. A potential set of future missions under study by NASA and ESA that would push these limits are those proposed for the Mars Sample Return

(MSR) campaign (see Figure 1). The current notional MSR campaign architecture consists of two follow-up missions to Mars 2020, consisting of a Sample Return Lander (SRL) mission, and an Earth Return Orbiter (ERO) mission. SRL would land on Mars with a fetch rover to retrieve the Mars 2020 samples and place them into Mars orbit within an Orbiting Sample (OS) container. The ERO would robotically capture the OS and return it to Earth within an Earth Entry Vehicle (EEV) using a Capture, Contain, and Return System (CCRS) [18] [19] [20]. The MSR campaign is a clear example of the level of complexity that robotic space missions have grown to encompass. Consequently, the complexity of space missions is quickly growing faster than the institution's ability to manage them [21]. If NASA and JPL are to succeed in future robotic space missions like those associated with Mars Sample Return, new systems engineering approaches to manage the growth in complexity associated with these future missions will be critical in order to control costs, maintain schedule, and ensure mission success.

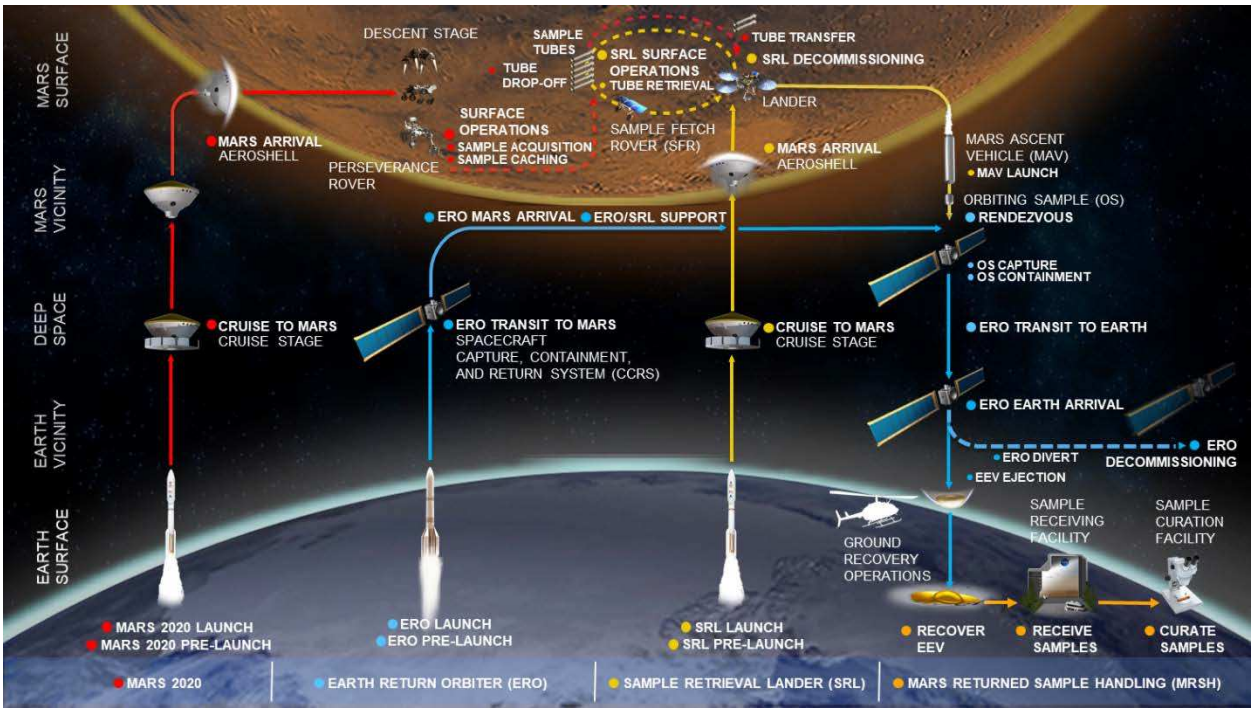


FIGURE 1. NOTIONAL MSR ARCHITECTURE. NOTE THAT ALL ELEMENTS BEYOND MARS 2020 ARE CONCEPTUAL [22].

Model-based Systems Engineering (MBSE) offers techniques to aid in the development of these types of complex systems by aiming to reduce design errors, reduce cost through prevention of costly rework, and improve system quality and project performance over traditional systems engineering techniques [23] [24]. MBSE seeks to improve on the state of the art in systems engineering through enhancing communications to aid in system presentation and understanding, reducing development risk through enabling ongoing requirements V&V and more accurate cost estimation, improving system quality in terms of requirements and requirements traceability, increasing productivity with systems engineering activities, and enhancing knowledge transfer through more effective capture of domain knowledge in a standardized form [25]. Additionally, MBSE places a focus on systems engineering activities in the earlier stages of the project life cycle, which aim to reduce the risk of the accruing magnified costs associated with dealing with defects detected later in the project life cycle that could trigger design changes. The aerospace industry in particular has high potential for overall project cost reductions by moving from traditional systems engineering to MBSE, based on the industry's tendency to develop projects with high system complexity, high environment complexity, and long lifespans, where these benefits would be more evident and enhanced relative to other industries [26]. Within the aerospace industry, robotic space systems feature close integration of large numbers of subsystems with highly integrated, complex, and intelligent structural and behavioral autonomous elements from multiple domains such as mechanical, electrical, control, and software [27] [28]. An MBSE approach can help engineers deal with the complexity of these robotic systems, as well as perform the thorough analysis at the foundational systems level necessary to realize these systems [27] [28].

An MBSE approach can help deal with system complexity during the architecting process, relative to traditional, document-based approaches, through use of a single model to represent key aspects of the architecture in a complete manner. An MBSE approach uses a set of languages, tools, and methods to facilitate the architecture model development, navigation, communication, and analysis. MBSE languages, such as SysML, provide the constructs for capturing descriptive information about the system, including system elements, element types, and element relationships, as well as abstracting and displaying this information in a visual manner to aid in comprehension [25]. MBSE tools, such as Cameo Systems Modeler, provide capabilities to store, retrieve, modify, and remove the information content of the architecture, automatically generate views of the architecture tailored to unique stakeholder concerns, and simulate the architecture to support analysis and evaluation [29]. MBSE methods, such as the Model-based System Architecture Process (MBSAP), provide object-oriented design methods to architect the system through structured decomposition of the system into modular and manageable levels of complexity using object-oriented principles such as abstraction, encapsulation, modularity, generalization, aggregation, interface definition, and polymorphism [30].

Despite the proposed benefits of MBSE, its practice has yet to be widely adopted [31]. Some causes for low MBSE adoption include technical issues, cultural issues, and economic barriers [32]. Additional challenges described by practitioners from experience applying MBSE on recent JPL flight projects include burdens associated with simultaneous introduction of a new MBSE approach, tool development, and processes, as well as negative impacts on project work due to immature tools and lack of understanding of user workflow [33]. The SysML language, which is commonly used within an MBSE approach, has also not gained widespread adoption in robotics,

though research has presented SysML as a viable framework in which to model robotic systems [34]. There is still a lack of substantial and compelling evidence in the literature to demonstrate the value of MBSE for robotic space systems. Robotic space systems may have much to benefit from an MBSE approach due to their intrinsic complexity, particularly if it is implemented in the early phases of the project during system architecting.

The purpose of this research is, therefore, to investigate and document the benefits of MBSE for architecting robotic space systems, particularly with describing, developing, and evaluating the system architecture. The MSR campaign ERO mission CCRS Capture and Orient Module (COM) system will be used as a case study to demonstrate these benefits. The overall goal is to gather evidence to quantify the costs and benefits of adopting MBSE for architecting robotic space systems. This includes evidence that an MBSE approach can capture more abstract architectural knowledge to more accurately describe the architecture, lower the numbers of resources, types of resources, and interfaces to reduce the complexity of the architecture development process, and lower the number of steps required to generate and execute scenarios efficiently to more effectively evaluate the system architecture's operation.

2. BACKGROUND

Model-Based Systems Engineering (MBSE) is a systems engineering approach that makes an integrated system model the primary artifact for systems engineering activities, rather than documents, as in traditional, document-based systems engineering [25]. A modeling language, modeling method, and modeling tool is necessary to implement MBSE effectively [35]. Examples of modeling languages include Systems Modeling Language (SysML) [36], Unified Modeling Language (UML) [37], Object-Process Methodology language (OPM) [38], Modelica [39], and ARCADIA Domain Specific Modeling Language (DSML) [40]. Examples of modeling methods include STRATA [41], Harmony for Systems Engineering Methodology [42], Object-Oriented Systems Engineering Method (OOSEM) [25], SYSMOD [43], MagicGrid [44], MBSAP [30], CESAMES Systems Architecting Method [45], ARChitecture Analysis and Design Integrated Approach (ARCADIA) [46], and Model-Based methodology to support the Space System Engineering (MBSSE) [47]. Examples of modeling tools include No Magic Cameo Systems Modeler [29], IBM Rational Rhapsody [48], PolarSys Capella [49], and Vitech GENESYS [50].

These modeling languages, methods, and tools enable MBSE to offer the following products:

1. A **descriptive model** of the system with accurate and precise encoding of system design information [51] (e.g., requirements, design, design rationale, and interrelationships [25]) enabled by the rich syntax and semantics of modeling languages;
2. Rigorous modeling techniques that incorporate traditional systems engineering best practices to **develop** a central unambiguous, organized, and precise system model [52]; and

3. An analytical model of the system that allows execution and simulation of the system to analyze and **evaluate** the system design, enabled by the well-defined semantics of modeling languages that can be interpreted computationally, and modeling tools capable of understanding and processing the modeling languages [53].

The above MBSE products provide the following benefits over traditional, document-based systems engineering:

1. **Enhanced knowledge capture** [52] with the ability to represent system structure, data, and functions more precisely and explicitly in a multidimensional format [23] with less ambiguity [53];
2. **Reduced effort** to implement system development through increased productivity, reduced inefficiencies, and reduced lag in information flow [51] [54] [55] [24]; and
3. **Increased efficiency** in evaluating the system through reduced time to simulate system scenarios through reduced coding time [56] and simplified maintenance of scenario data [57].

Several recent spacecraft flight projects applied SysML to improve system design knowledge capture. Fosse et al. [58] aimed to improve accurate capture and communication of the Mars 2020 flight system design within a SysML model that served as an authoritative source of information. The Flight System Systems Engineering team created a SysML extension called the Mars2020 Modeling Framework that included flight project-specific model element definitions, nomenclature, stereotypes, patterns, and viewpoints. The modeling framework provided a precise, unambiguous description of the flight system design for the project team. The Europa Clipper project aimed to improve communication of desired system behaviors to software engineers that are typically poorly articulated as textual representations [21]. The systems

engineering team modeled the flight system design in MagicDraw and specified system behavior using SysML behavior diagrams. The system model was reported to more explicitly describe behavioral information and provide more stable and reliable estimates for key technical margins. The Asteroid Redirect Robotic Mission (ARRM) Systems Engineering team applied MBSE to capture the mission's operation timeline, system decomposition, and functional requirements [59]. These were modeled in SysML activity diagrams, block definition diagrams, and requirements diagrams, while incorporating a foundational ontology developed by JPL's Integrated Model Centric Engineering (IMCE) project-support community that formally names and defines types, properties, and relationships of entities that represent space missions. The team showed that MBSE can deliver enhanced system products over the project life cycle, as well as facilitate more rigorous system representation and definition than is typically possible in early mission concept development.

MBSE methods have aided in the development of complex space systems. The OSIRIS-Rex Science Processing and Operations Center (SPOC) Systems Engineering team explored a layered MBSE approach as a means of managing the development of the complex architecture of the SPOC [60]. The team used CORE to model the system. The layered MBSE approach performed requirements development, behavior analysis, architecture development, and verification and validation planning for each level before proceeding down to the next, lower level, starting with the top-most system level. The approach provided a means to design a more consistent, well-documented system. Mazzini et al. assessed the applicability of the MBSSE methodology on the ExoMars mission [47]. System requirements, system context, data, control flows, mission use cases, scenarios, functional architecture, and software architecture were modeled in progression during Phases 0, A, and B of the project. The MBSSE approach proved successful in defining a

preliminary space system and offered improved traceability and separation of concerns between systems engineering and software engineering. Estable et al. applied a Federated and Executable Models MBSE methodology during the architecture definition phase of the ESA e.Deorbit robotic satellite mission study with the aim to maintain systems thinking over the complete systems engineering process [61]. SysML models were developed to capture the mission CONOPS, system capabilities, functional architecture, safety diagram, fault tree, product tree, and requirements using Cameo Systems Modeler. The authors reported the methodology to be an adequate systems engineering method, to maintain systems thinking, and to show potential to lead to higher efficiency in systems engineering work.

Multiple research efforts demonstrated the capabilities of executable SysML models for performing system analysis, system evaluation, and requirements verification and validation. Karban et al. [57] investigated the ability to perform system analysis through executable SysML models for requirements verification. They employed an extension to OOSEM called the Executable System Engineering Method (ESEM) for scenario-based power analysis of the Thirty Meter Telescope using the Cameo Simulation Toolkit [62] to verify power requirements of the telescope. They demonstrated that the method was capable of carrying out the complex analysis needed to show that the system satisfies its peak power requirement. The Radio Aurora Explorer (RAX) CubeSat project aimed to demonstrate the applicability of MBSE for modeling mission operations, focusing on the power and communication subsystems [63]. An executable system model was developed using SysML block definition diagrams, requirement blocks, parametric diagrams, activity diagrams, and state machines. Missions were simulated using Cameo Simulation Toolkit to execute top-level activity diagrams, which in turn called engineering analysis models integrated in ModelCenter to generate time-history energy and data download

states. The approach demonstrated the potential to simulate missions accurately through executable SysML behavioral diagrams. Gregory et al. analyzed the Payload Data Handling Unit of the Biomass spacecraft to validate spacecraft memory allocation [64]. A high-level functional design of the spacecraft, along with a set of requirements, were modeled in SysML and simulated over a 70-day mission profile to quantify performance in terms of memory usage. The MBSE analysis approach resulted in a more streamlined, consistent, and traceable approach, and showed that an executable system model can provide early validation of the functional architecture.

The above case studies demonstrated feasibility of MBSE and highlighted benefits that MBSE brings in describing, developing, and evaluating complex space systems. However, none of these studies performed in-depth comparative quantitative analysis of applying MBSE vs. non-MBSE approaches to provide sufficient, direct, quantitative evidence of MBSE's advantages over traditional, non-MBSE approaches. A literature review by Carroll and Malins [24] found no case studies that compared an MBSE approach side-by-side with a non-MBSE (document-based) approach. Additionally, White and Mesmer [54] point to a lack of MBSE case studies and success stories as a barrier to more widespread adoption, and Carroll and Malins [24] mention that further study is warranted to establish a definitive ROI for implementing MBSE. Also, despite there being a large number of SysML publications since 2005, these publications are primarily dominated by production, aerospace, and mechatronic applications, with the robotics application domain showing a smaller presence in the literature [65]. This research aims to provide additional robotic space system case studies with more direct, quantitative proof of the advantages of MBSE relative to non-MBSE approaches in order to help promote its use in the robotics application domain, specifically with robotic space systems.

Presenting evidence of the relative advantage of MBSE over non-MBSE approaches within the robotic space systems domain can be an important contribution to increase its rate of adoption within that community. Diffusion of innovations theory [66] posits that potential adopters base their decision to adopt an innovation according to:

1) The innovation's perceived high relative advantage over the idea it supersedes,
2) High compatibility,
3) Low complexity related to its difficulty to understand and use,
4) High trialability,
5) High observability of results [66].

Currently, there are still perceptions within the systems engineering community that challenge faster rates of adoption of MBSE. Some of these include:

- Lack of value of MBSE, and that the costs of MBSE might outweigh the benefits [54] [55] (creating perceptions of low relative advantage of MBSE),
- Needed shift in mindset, tools, language, and methods to implement MBSE [33] (creating perceptions of low compatibility of MBSE with the cultural environment),
- Highly complex MBSE languages [67] and uncertainty of what MBSE is and how it can be used in practice [54] (creating perceptions of high complexity of MBSE associated with difficulty in understanding and use),
- Substantial upfront investment to get started with MBSE [26] [68] and limited 'out-of-the-box' integration capability of MBSE tools [69] (creating perceptions of low trialability), and
- Gains associated with MBSE (e.g., cost savings, defect reduction, and rework prevention [24]) not realizable until the latter stages of the system life cycle [26] and with difficultly

in observing and communicating its benefits due to their abstract nature (creating perceptions of low observability).

The above perceptions all contribute to the overall challenge that MBSE faces with adoption. Acquiring additional evidence of the relative advantage of MBSE over non-MBSE approaches addresses one of the key challenges to MBSE adoption. Even if the advantages of MBSE can be made more explicit, the evidence must be significant enough so that the perceived benefits of MBSE clearly outweigh the perceived costs. Potential adopters' resistance to change is explained by research showing that experienced and expert individuals commonly stick with preferred, well-practiced, generic procedures that provide fast, incremental feedback and require less cognitive effort, even when more efficient, globally optimal procedures exist [70]. The purpose of this research is to acquire evidence of the benefits of MBSE approaches over traditional, non-MBSE approaches for architecting robotic space systems though comparative analysis, focusing on quantitative evidence that MBSE better describes, develops, and evaluates the system architecture, with the goal to contribute to the adoption of MBSE within the robotics space systems domain.

3. PROBLEM FORMULATION

Based on a review of the literature, there is an absence of case studies providing side-by-side comparisons of projects using both MBSE and traditional, non-MBSE approaches with quantitative evidence that measures the relative advantage of one approach over the other. In particular, there is a lack of quantitative comparative analysis studies of MBSE applied to robotic space systems during the architecture phase of the project, where MBSE promises to provide the most value. This leads to the following overarching question:

What measurable advantages does MBSE provide to architecting robotic space systems from a knowledge capture and process implementation viewpoint?

Three areas where MBSE can aid in architecting complex systems will be investigated in this research effort to answer the overarching question:

1) **Can an MBSE approach better capture the information content required to describe a robotic space system architecture relative to a non-MBSE approach, as assessed by a higher categorized fraction of architecture content that can be formally captured in the appropriate knowledge category with the language and tool?**
2) **Can an MBSE approach reduce the implementation effort required to develop a robotic space system architecture relative to a non-MBSE approach, as assessed by a higher quantity of information transfer between tasks that can be automated for carrying out the architecting process?**
3) **Can an MBSE approach more efficiently evaluate a robotic space system architecture than a non-MBSE approach, as assessed by a higher quantity of**

knowledge that can be automatically processed during modeling and simulation activities for system requirements verification?

In order to investigate the above research questions, a representative robotic space system with high complexity was chosen as a case study. The system chosen was the Capture and Orient Module (COM), which was developed at the NASA Jet Propulsion Lab for the future proposed Mars Sample Return campaign.

The Capture and Orient Module (COM), shown in Figure 2, performs the initial operations of the Capture, Contain, and Return System (CCRS) for the Mars Sample Return (MSR) Earth Return Orbiter (ERO) mission, as depicted in Figure 1. The COM operations include capturing, constraining, orienting, inspecting, and assembling the Orbiting Sample (OS) into the Primary Containment Vessel (PCV) in preparation for PCV sealing and installation into the Earth Entry Vehicle (EEV) for eventual delivery to Earth. Various architectures for OS capture systems have been studied and proposed over the past 20 years [71] [72] [73] [74] [75]. The concept for the COM referred to in this research is shown in Figure 3 and Figure 4.

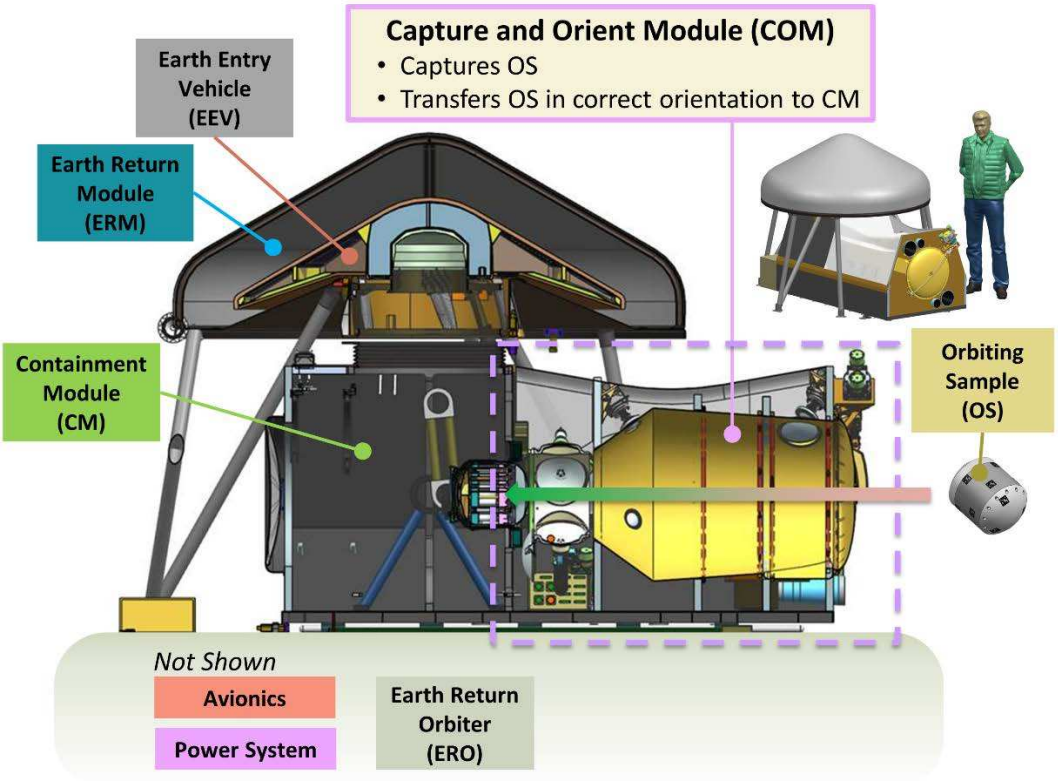


FIGURE 2. NOTIONAL CAPTURE, CONTAINMENT, AND RETURN SYSTEM CONCEPT.

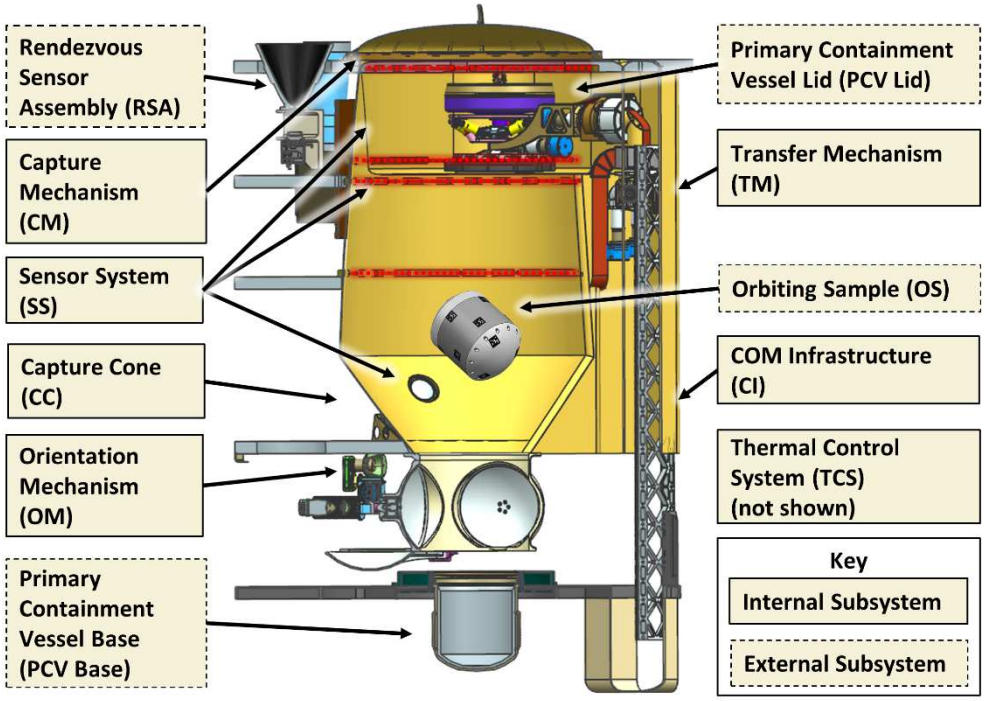


FIGURE 3. NOTIONAL CAPTURE AND ORIENT MODULE CONCEPT.

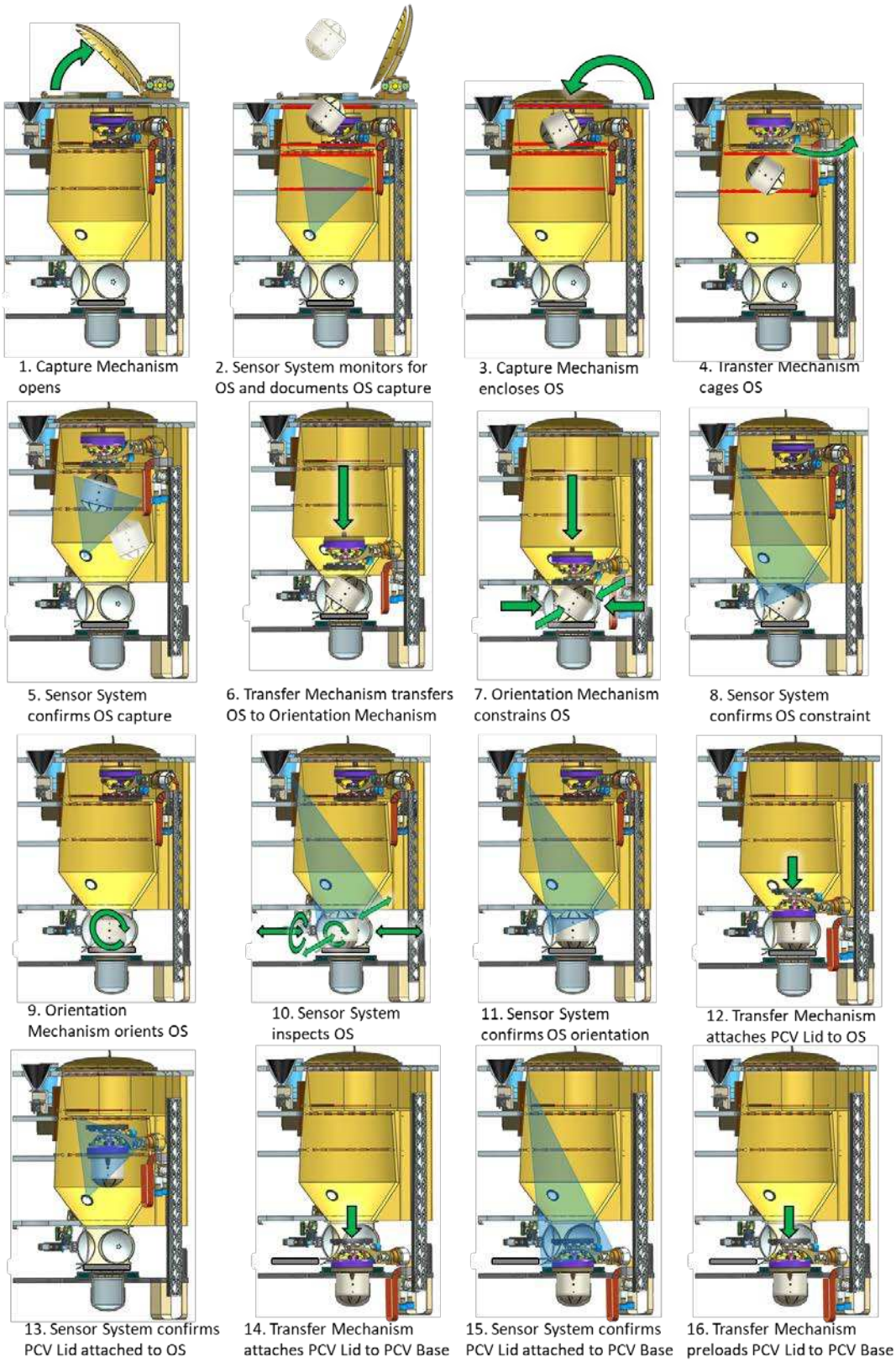


FIGURE 4. CAPTURE AND ORIENT MODULE OPERATIONAL CONCEPT.

The primary COM operations take place in Mars orbit. Certain operations, such as OS capture, rely on event-driven sequences that are impossible or impractical to initiate from mission control due to the communication delay that occurs with data transmission between Earth and Mars. Therefore, a level of autonomy will be required for the COM to achieve its goals while operating independent of ground control. This autonomy can be in the form of pre-planned sets of instructions transmitted to the spacecraft and executed by the CCRS Command and Data Handling (C&DH) system [76].

Because the COM relies on integrated mechanical, electrical, and software systems to perform autonomous operations, it is considered a robotic system. Additionally, the COM architecture was classified along the four architecture taxonomic dimensions defined in [30]:

- Abstraction: Progression of the architecture from more abstract to more concrete, ranging from the system context to the physical implementation
- Organization: Level of decomposition of the architecture, ranging from an enterprise or system-of-systems level down to its individual components
- Categorization: Architectural categories that may possess unique system characteristics, follow particular business processes, serve specific operational uses, or meet particular stakeholder requirements
- Time: Period in the system life cycle during which the architecture is defined and has particular rules associated with its architectural evolution

The architecture space for the first three of the architecture taxonomic dimensions used to classify the COM architecture is shown in Figure 5. The axis of organization spans from the top-level Mars Sample Return Campaign, within which the system participates, down to the individual system components. The axis of categories defines the key functional domains

relevant to the CCRS project and key to its architecture. The COM architecture scope addresses the module level down to its components and spans through operational to physical definition. The COM architecture categories include the following functional domains:

- Robotics (kinematics, workspace analysis, manipulation, pose estimation)
- Mechanical (structures, mechanisms, sensors)
- Electrical (power, data, harness)
- Thermal (temperature monitoring and control)
- Flight software (high-level behavior, low-level control)
- Planetary Protection (protecting Earth from exposure to unsterilized Mars particles)

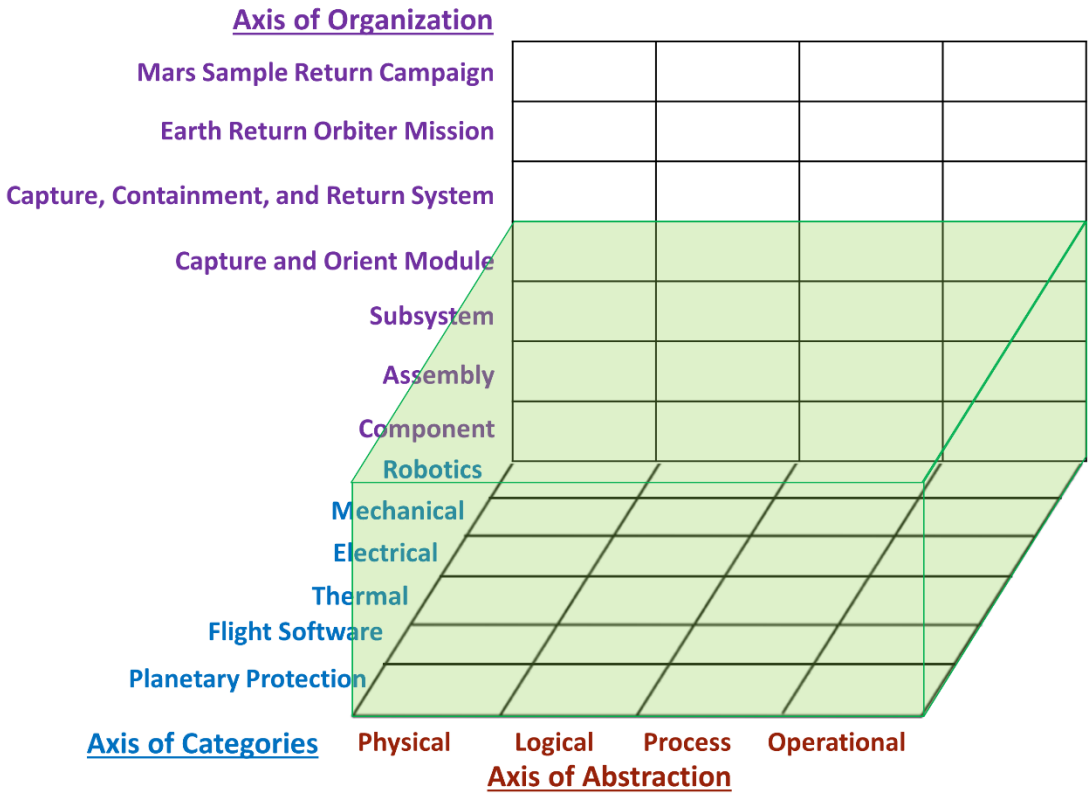


FIGURE 5. CAPTURE AND ORIENT MODULE ARCHITECTURE CLASSIFICATION SHOWN IN GREEN ALONG THE AXES OF ABSTRACTION, ORGANIZATION, AND CATEGORIZATION BASED ON THE ARCHITECTURE TAXONOMY DESCRIBED IN [30].

The COM architecture time dimension was defined by the seven phases of the NASA Project Life-Cycle (see Figure 6). Architecting of an initial, feasible concept of the COM takes place during Pre-Phase A. In Pre-Phase A, the project looks at a range of ideas and alternative architectures, determines the feasibility of the desired system, and develops candidate system concepts [77].

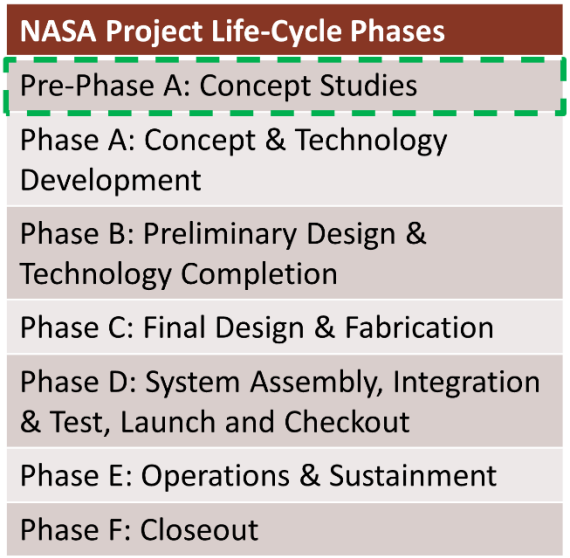


FIGURE 6. CAPTURE AND ORIENT MODULE ARCHITECTURE PHASE BOXED IN GREEN WITHIN THE NASA PROJECT LIFE-CYCLE BASED ON NASA PROCEDURAL REQUIREMENTS NPR 7123.1B [78].

The scope of the architecture was limited for this research to fit within the available project resource constraints (i.e. workforce availability, schedule) and operate in an early project environment where many system elements are still ill-defined:

- Single reference architecture for a candidate system concept (though trades and alternatives at subsystem levels and decision rationale for the selected approaches will be captured)
- Primarily the hardware aspects of the robotic system (the software and avionics aspects of the architecture will be developed as separate modules within CCRS)

- Interaction (type B) scenarios that focus primarily on the direct interactions between the system of interest and the actors and external systems within the system context for the top-level COM scenarios [79]
- Primary portion of the COM main scenario that starts from OS capture and ends with OS assembly into the PCV (development of commissioning, decommissioning, and support services will be developed at later phases of the project life cycle)

The next three chapters perform comparative analyses of MBSE and Non-MBSE approaches to investigate the benefits of MBSE in describing, developing, and evaluating complex robotic space system architectures, using the COM Pre-Phase A architecture development as a case study. Chapter 4 explores Research Question 1, assessing how MBSE increases information capture of the COM architecture descriptive model. Chapter 5 explores Research Question 2, assessing how MBSE reduces implementation effort during the COM architecting process. Chapter 6 explores Research Question 3, assessing how MBSE improves evaluation efficiency when modeling and simulating the COM architecture.

4. COMPARATIVE ANALYSIS OF AN MBSE APPROACH TO A TRADITIONAL SE APPROACH FOR ARCHITECTING A ROBOTIC SPACE SYSTEM THROUGH KNOWLEDGE CATEGORIZATION[1]

4.1. Introduction

Space missions are considered high-risk systems with tightly coupled subsystems and complex interactions. Due to the multidisciplinary nature of space missions, they exhibit complexity in requirements, design, flight software development, testing, and operations, which has been correlated to higher spacecraft cost and lower rates of mission success [15]. Robotic space systems in particular, are becoming increasingly more complex, and hardware capability and software complexity is expected to continue to grow [16]. If this trend in complexity growth continues, higher spacecraft costs and mission success risk can also be expected to grow.

Model-based Systems Engineering (MBSE) offers techniques to aid in the development of complex systems by aiming to reduce design errors, reduce cost through prevention of costly rework, and improve system quality and project performance over traditional systems engineering techniques [23] [24]. Using MBSE reduces the risk of late design defects by placing a greater focus on early conceptual design and preliminary design stages of a project [26]. The cost associated with fixing errors increases the later they are discovered in the design process. Therefore, a project can achieve its greatest cost savings by finding errors early in the project (or avoiding them in the first place) [80]. The aerospace industry in particular has high potential for

[1] This chapter previously appeared as an article in the *Systems Engineering*. The citation is as follows: Younse, P., J. Cameron, and T.H. Bradley, "Comparative Analysis of a Model-based Systems Engineering Approach to a Traditional Systems Engineering Approach for Architecting a Robotic Space System through Knowledge Categorization," *Systems Engineering*, 2021.

overall project cost reductions by moving from traditional systems engineering to MBSE based on the industry's tendency to possess projects with high system complexity, high environment complexity, and long lifespans, where these benefits would be more evident and enhanced relative to other industries [26]. Within the aerospace industry, robotic space systems feature close integration of large numbers of subsystems with highly integrated, complex, and intelligent structural and behavioral autonomous elements from multiple domains such as mechanical, electrical, control, and software [27] [28]. An MBSE approach can help engineers deal with the complexity of these robotic systems, as well as perform the thorough analysis at the foundational systems level necessary to realize these systems [27] [28].

An MBSE approach addresses system complexity during the architecting process through the development of a single model to represent key aspects of the architecture in a complete manner. An MBSE approach uses a set of languages, tools, and methods to facilitate the architecture model development, navigation, communication, and analysis. MBSE languages, such as the SysML used in this research, provide the constructs for capturing descriptive information about the system, including system elements, element types, and element relationships, as well as abstracting and displaying this information in a visual manner to aid in comprehension [25]. MBSE tools, such as the Cameo Systems Modeler used in this research, provide capabilities to store, retrieve, modify, and remove the information content of the architecture, automatically generate views of the architecture tailored to unique stakeholder concerns, and simulate the architecture to support analysis and evaluation [29]. MBSE methods, such as the Model-based System Architecture Process (MBSAP) used in this research, provide object-oriented design methods to architect the system through structured decomposition of the system into modular and manageable levels of complexity using object-oriented principles such as abstraction,

encapsulation, modularity, generalization, aggregation, interface definition, and polymorphism [30].

Return-on-investment (ROI) provided by MBSE is commonly measured in terms of development cost, development time, and number of defects [24]. The purpose of this research is to investigate the value of MBSE in addition to these commonly presented measures of ROI, particularly in describing the system architecture by better capturing a larger quantity of architecture knowledge than traditional, document-based systems engineering approaches. The Mars Sample Return (MSR) campaign Earth Return Orbiter (ERO) mission Capture, Containment, and Return System (CCRS) Capture and Orient Module (COM) system was used as a case study to demonstrate these benefits.

4.2. Background

The following section provides an overview of MBSE, system architecting, high-level architecture knowledge, the Capture and Orient Module robotic space system used as a case study in this research, and the overall research purpose.

4.2.1. Model Based Systems Engineering

Traditional systems engineering uses documents as the primary artifacts of design [25]. The NASA Space Flight Program and Project Management Handbook provides a list of configuration-controlled and non-configuration-controlled documents commonly used in NASA flight projects [81]. These documents may include spreadsheets, presentations, and manuscripts. Spreadsheets contain tabular sheets that store both analog and textual data within individual cells of the sheets. Presentations contain slides to visually organize and represent images and text on a canvas. Manuscripts capture textual and visual information in a standard manuscript format. Common tools used to generate spreadsheets, presentations, and manuscripts include Microsoft

Excel, Microsoft PowerPoint, and Microsoft Word [82]. Other tools used to help both generate and manage traditional systems engineering artifacts include IBM Rational DOORS NG [83], Siemens Teamcenter [84], and Atlassian Confluence [85].

MBSE is a systems engineering methodology that makes an integrated system model the primary artifact for systems engineering activities. A modeling language, modeling method, and modeling tool are necessary to implement MBSE effectively [35]. Examples of modeling languages include Systems Modeling Language (SysML) [36], Unified Modeling Language (UML) [37], Object-Process Methodology (OPM) [38], Modelica [39], and ARChitecture Analysis and Design Integrated Approach Domain Specific Modeling Language (DSML) [40]. Examples of modeling methods include STRATA [41], Harmony for Systems Engineering Methodology [42], Object-Oriented Systems Engineering Method (OOSEM) [25], Systems Modeling Process (SYSMOD) [43], MagicGrid [44], Model-Based System Architecture Process (MBSAP) [30], CESAMES Systems Architecting Method [45], Architecture Analysis and Design Integrated Approach (ARCADIA) [86], and Model-Based methodology to support the Space System Engineering (MBSSE) [47]. Examples of modeling tools include No Magic Cameo Systems Modeler [29], IBM Rational Rhapsody [48], PolarSys Capella [49], and Vitech GENESYS [50].

These modeling languages, methods, and tools enable MBSE to generate a descriptive model of the system with accurate and precise encoding of system design information [51] (e.g., requirements, design, design rationale, and interrelationships [25]) enabled by the rich syntax and semantics of modeling languages. This provides the benefit of enhanced knowledge capture [52], with the ability to represent system structure, data, and functions more precisely and

explicitly in a multidimensional format with less ambiguity, relative to traditional, document-based systems engineering [23] [53].

Several recent spacecraft flight projects applied SysML to improve system design knowledge capture. Fosse et al. [58] aimed to improve accurate capture and communication of the Mars 2020 flight system design within a SysML model that served as an authoritative source of information. The flight system systems engineering team created a SysML extension called the Mars2020 Modeling Framework that included flight project-specific model element definitions, nomenclature, stereotypes, patterns, and viewpoints. The modeling framework provided a precise, unambiguous description of the flight system design for the project team. The Europa Clipper project used SysML to improve the communication of system behaviors to software engineers [21]. The systems engineering team modeled the flight system design in MagicDraw and specified system behavior using SysML behavior diagrams. The system model was reported to more explicitly describe behavioral information and provide more stable and reliable estimates for key technical margins. The Asteroid Redirect Robotic Mission (ARRM) systems engineering team applied MBSE to capture the mission's operation timeline, system decomposition, and functional requirements [59]. These were modeled in SysML activity diagrams, block definition diagrams, and requirements diagrams, while incorporating a foundational ontology developed by JPL's Integrated Model Centric Engineering project-support community that formally names and defines types, properties, and relationships of entities that represent space missions. The team showed that MBSE can deliver enhanced system products over the project life cycle, as well as facilitate more rigorous system representation and definition than is typically possible in early mission concept development.

The above case studies demonstrated feasibility of MBSE and highlighted benefits that MBSE brings in describing complex space systems. However, few of these studies provide quantitative evidence of MBSE's advantages over traditional, non-MBSE approaches. A literature review by Carroll and Malins [24] found no case studies that compared an MBSE approach side-by-side with a non-MBSE (document-based) approach. White and Mesmer [54] point to a lack of MBSE case studies and success stories as a barrier to more widespread adoption, and Carroll and Malins [24] mention that further study is warranted to establish a definitive return on investment for implementing MBSE. Despite there being a large number of SysML publications since 2005, these publications are primarily dominated by production, aerospace, and mechatronic applications, with the robotics application domain showing a lesser presence in the literature [65]. This research focuses on exploring the benefits of MBSE within the robotics application domain. Robotic systems are distinguished from mechatronic systems in that robotic systems consist of mechanical, electrical, controls, and software subsystems (e.g., perception and autonomy), while mechatronic systems only consist of mechanical, electrical, and controls subsystems [87]. Additionally, aerospace systems are distinguished from robotic systems in that aerospace systems typically consist of atmospheric and space vehicles [88], while robotic systems typically consist of robotic platforms and robotic payloads transported via the space vehicle [87].

4.2.2. System Architecting

Architecture is the fundamental concepts or properties of a system in its environment embodied in its elements, relationships between those elements, and principles of their design and evolution [89]. System architecting is the process carried out to synthesize the system architecture. Examples of system architecting processes include the synthetic process described

by Crawley et al. [90], system architecting process described by Weikiens et al. [91], the architecture definition process described by INCOSE [92], the Model-Based System Architecture Process (MBSAP) described by Borky and Bradley [30], the Architecture Design and Evaluation Process described by Min and Noguchi [93], and the System Architecting and Design Space Characterization Process described by Raz et al. [94].

In addition to system architecting processes, system architecting is commonly guided by architecture frameworks [30]. An architecture framework collects and relates viewpoints to enable the system architect to construct useful and consistent architecture descriptions [95]. Examples of architecture frameworks include The Open Group Architecture Framework (TOGAF) [96], US Department of Defense Architecture Framework (DoDAF) [97], Zachman Framework [98], CESAM Framework [45], Model-Based System Architecture Process (MBSAP) Framework [30], and MagicGrid Framework [44].

Maier and Rechtin describe architecting to be an integration of competing subsystems and interests [99]. They also stress the importance of a single, shared vision of the system amongst the architecting team. Similarly, the ISO/IEC/IEEE 42010 standard asserts that the architecture of a system is a holistic conception of that system's fundamental properties, which is best understood through multiple architectural views [89]. With traditional, document-based systems engineering, it is difficult to assess the completeness and consistency of the system because information is spread across various documents [100]. Therefore, it is difficult to create and maintain a single, shared vision of the system amongst the team. MBSE, however, through its use of a single system model to generate views, helps to maintain consistency amongst all its views and create a single, shared vision of the system amongst the team.

4.2.3. High-Level Knowledge in System Architecture

An architecture description is a set of artifacts that captures and communicates knowledge about the system architecture to all stakeholders in order to meet their needs for information and insights [30]. Knowledge categories of particular value for architecting highly complex systems are those associated with higher-level, abstract categories of knowledge, such as architecting methods, experience-based heuristics, abstractions, integrated models, organizational concepts, component interfaces, governing design principles, and design decisions [99] [100] [101]. These types of knowledge represent and communicate deep understanding amongst subject matter experts, as well as reflect how the expert knowledge is organized and applied to solve problems within a particular system context [102].

For example, the Jet Propulsion Laboratory (JPL) documents and implements high-level governing design principles through its JPL Design Principles, which specify essential attributes of JPL space flight systems that contribute to robust design, verification/validation, and operation based on lessons learned from past flight project experience [103] [104]. One example of a JPL Design Principle is "Visibility of Spacecraft Status," which states that a spacecraft information system shall provide telemetry data to the ground to support rapid assessment of the spacecraft's statuses under normal, stressed, and faulted operations [103]. This design principle stems from a NASA lesson learned from a scan platform anomaly that occurred during the Galileo mission, where mission analysts were unable to diagnose the anomaly because the spacecraft architecture did not provide for revolving, short-term storage of downlink data to record telemetry associated with the anomaly [105]. The rationale for including this design principle in a spacecraft architecture is that storing telemetry data preceding and following a spacecraft fault could potentially assist mission analysts on the ground with fault diagnosis and recovery efforts.

This “Visibility of Spacecraft Status” design principle was incorporated in the Mars Science Laboratory (MSL) rover avionics architecture. On Sol-200 of the MSL mission, the rover encountered uncorrectable errors in the NAND flash memory that led to an inability of the prime computer to turn off for its normal recharge session, which could have led to a mission-catastrophic event [106]. Because the avionics architecture provided telemetry data of the anomaly, engineers at JPL were able to successfully diagnose the issue and recover from the potentially mission-ending anomaly. Note that design principles are distinguished from requirements, in that design principles are general concepts recommended for space flight systems to follow, whereas system requirements are specific specifications that must be implemented onto a flight system. Design principles can be instantiated as system requirements onto a specific flight system if deemed appropriate.

Vincenti [107] proposes additional examples of relevant higher-level engineering design knowledge, which include fundamental design concepts, technical device criteria, theoretical tools to carry out design, and practical considerations derived from experience in practice. An example of a practical consideration relevant to robotic space systems is the use of brushless direct current (DC) motors over brushed DC motors for long-duration missions in vacuum environments [108] [109]. This consideration stems from observations of accelerated brush failure in DC motors used in high-altitude rotating equipment in World War II and satellites in the early years of space exploration [110]. The MSL project recommended brushless DC motors for its rover mechanisms as part of the MSL rover architecture to ensure long actuator life based on this practical consideration [7]. As of summer of 2020, MSL rover mechanisms running off brushless DC motors have been operating for over four Mars years (eight Earth years), much longer than the prime mission of one Mars year (two Earth years). This success can be attributed

to the fact that MSL was able to capture and incorporate these types of practical knowledge into its architecture development process.

The above examples demonstrate how higher-level knowledge, such as design principles and practical considerations, contribute to robust system architectures. It is these abstract principles, design patterns, and rules that build the architecture and guide its development. Accordingly, a robust architectural approach should be capable of capturing this high-level knowledge to ensure they are properly implemented in the final system design. In the system architecture framework used in this study, design principles applied to system elements were captured as "Applicable Design Principles" within the system element specifications, and selected architectural approaches based on practical considerations were captured as "Recommended Approaches" and "Approach Rationales" within the element specifications. Additional high-level knowledge deemed important were also captured within the architecture framework, such as conceptual structures that describe interrelationships between system elements, and metacognitive strategies that describe system development strategies.

4.2.4. Application to Robotic Space Missions

This research focuses on the architecture of the Capture and Orient Module (COM), which is currently in development for the future proposed Mars Sample Return campaign [22]. The COM performs the initial operations of the Capture, Containment, and Return System (CCRS) for the Mars Sample Return (MSR) Earth Return Orbiter (ERO) mission (see Figure 7). These include capturing, constraining, orienting, inspecting, and assembling the Orbiting Sample (OS) into the Primary Containment Vessel (PCV) in preparation for PCV sealing and installation into the Earth Entry Vehicle (EEV) for delivery to Earth. Various architectures for OS capture systems

were proposed and studied over the past 20 years [71] [72] [73] [74] [75]. An early concept for the COM is shown in Figure 8. The COM consists of the following subsystems:

- Capture Mechanism to contain any unsterilized Martian dust arriving with the OS on its surface
- Sensor System to trigger closure the Capture Mechanism and Transfer Mechanism during OS capture, document the capture event, confirm OS capture and orientation, and inspect the surface of the OS
- Capture Cone to catch and contain the OS
- Orientation Mechanism to constrain and orient the OS
- Transfer Mechanism to cage the OS, transfer the OS through the COM subsystems, assemble the OS into the PCV, and maintain preload on the PCV Lid during sealing
- COM Infrastructure to integrate the COM elements into the Capture and Containment Module (CCM)

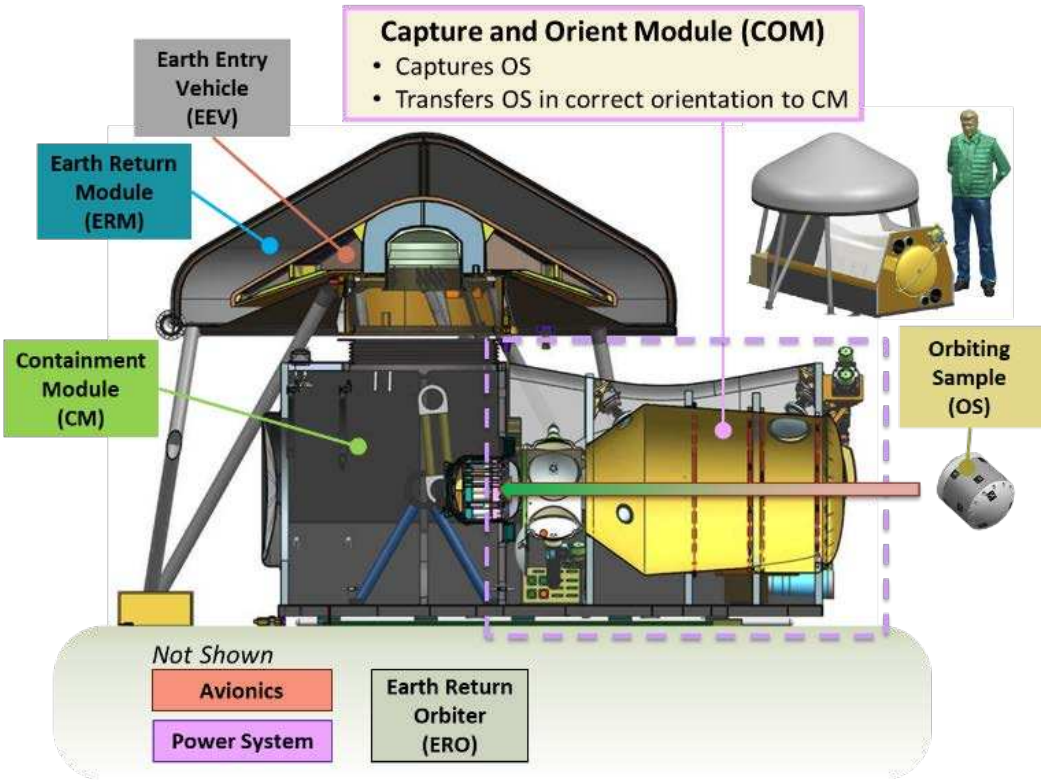


FIGURE 7. NOTIONAL CAPTURE, CONTAINMENT, AND RETURN SYSTEM (CCRS) CONCEPT [22].

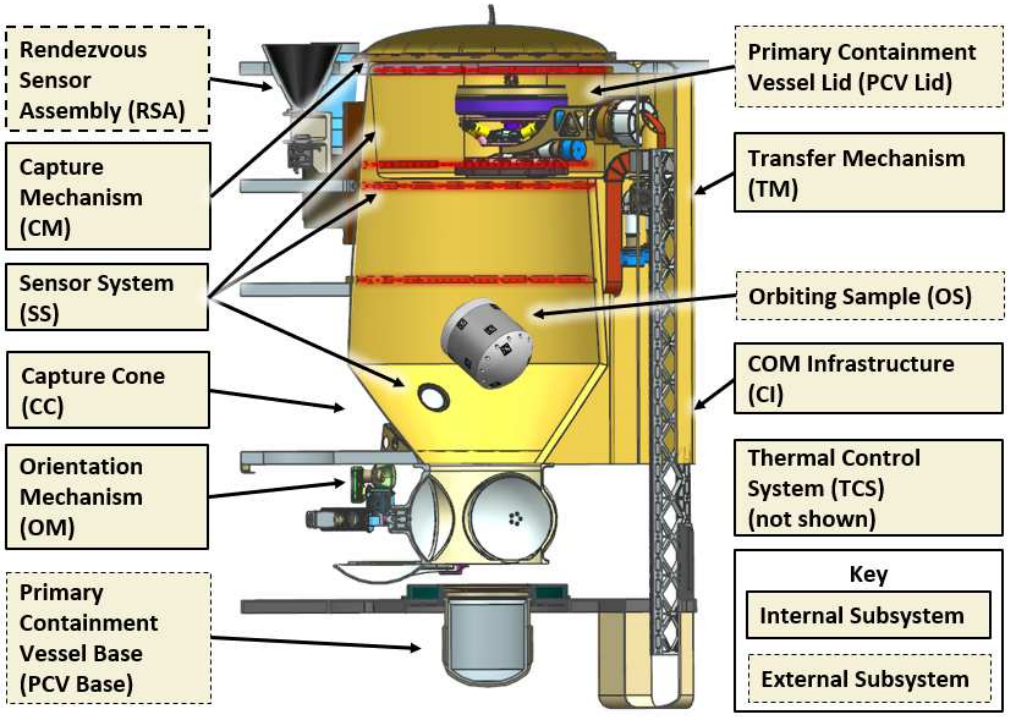


FIGURE 8. NOTIONAL CAPTURE AND ORIENT MODULE (COM) CONCEPT [22].

The primary COM operations take place in Mars orbit. Certain operations, such as OS capture, rely on event-driven sequences that are impossible to initiate from mission control due to the communication delay that occurs with data transmission between Earth and Mars. Therefore, a level of autonomy is required for the COM to achieve its goals while operating independent of ground control. This autonomy could be in the form of pre-planned sets of instructions transmitted to the spacecraft and executed by the CCRS Command and Data Handling (C&DH) system [76].

The COM architecture was classified along the four architecture taxonomic dimensions defined in [30]:

- Abstraction: Progression of the architecture from more abstract to more concrete, ranging from the system context to the physical implementation
- Organization: Level of decomposition of the architecture, ranging from an enterprise or system-of-systems level down to its individual components
- Categorization: Architectural categories that possess unique system characteristics, follow particular business processes, serve specific operational uses, or meet particular stakeholder requirements
- Time: Period in the system life cycle during which the architecture is defined and has particular rules associated with its architectural evolution

The architecture space for the abstraction, organization, and categorization dimensions used to classify the COM architecture is shown in Figure 9. The axis of organization spans from the top-level Mars Sample Return Campaign, within which the system participates, down to the individual system components. The axis of categories defines the key functional domains relevant to the CCRS project and key to its architecture. The COM architecture scope addresses

the module level down to its components and spans through operational to physical definition. The COM architecture categories include the following JPL-defined functional domains:

- Robotics (kinematics, workspace analysis, manipulation, vision)
- Mechanical (structures, mechanisms, sensors)
- Electrical (power, data, harness)
- Thermal (temperature monitoring, heaters, thermal control)
- Flight software (high-level behavior, low-level control)
- Contamination Control/Planetary Protection (controlling organic and inorganic materials and processes, protecting Mars from contamination of Earth life, protecting Earth from exposure to unsterilized Mars particles)

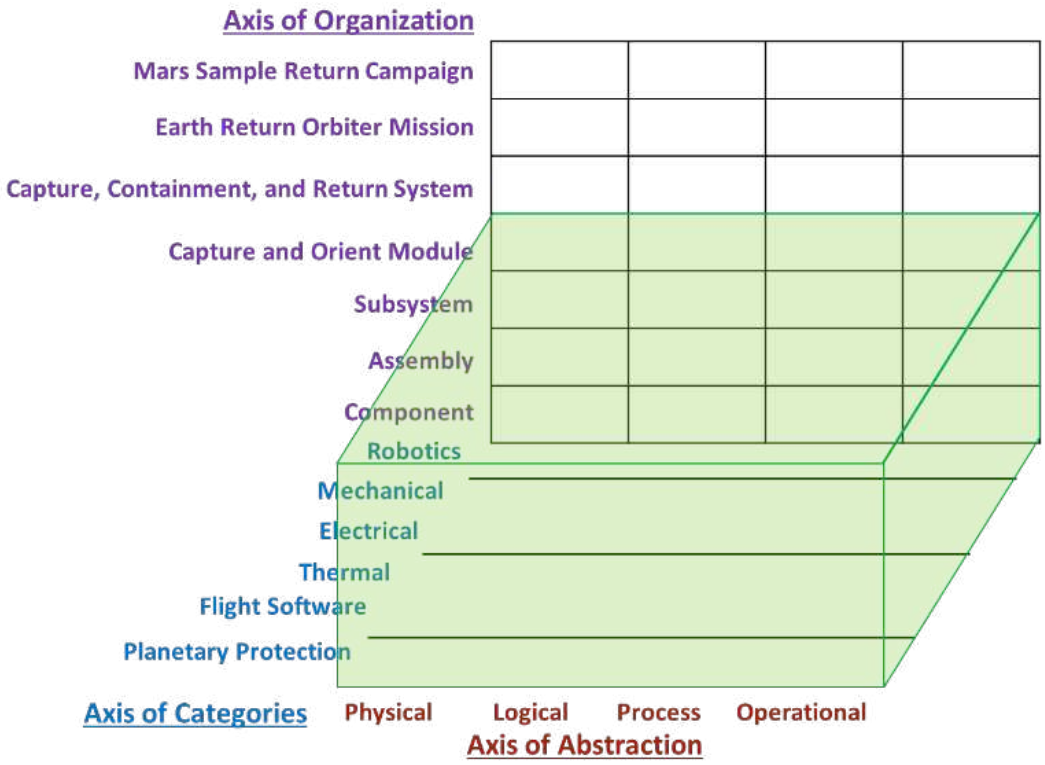


FIGURE 9. CAPTURE AND ORIENT MODULE ARCHITECTURE CLASSIFICATION SHOWN IN GREEN ALONG THE AXES OF ABSTRACTION, ORGANIZATION, AND CATEGORIZATION BASED ON THE ARCHITECTURE TAXONOMY DESCRIBED WITHIN THE MBSAP METHODOLOGY [30].

Note that the robotics domain architecture category is distinguished from the COM robotic system itself, in that the robotics domain is the subject area concerned with knowledge associated with robotics-specific subject matters (e.g., kinematics, workspace analysis, manipulation, vision), whereas the robotic system is the system itself within which the robotics domain participates.

The COM architecture time dimension was defined by the seven phases of the NASA Project Life-Cycle (see Figure 10) [78]. Architecting an initial, feasible concept of the COM takes place during Pre-Phase A. In Pre-Phase A, the project looks at a range of ideas and alternative architectures, determines the feasibility of the desired system, and develops candidate system concepts [77].

FIGURE 10. CAPTURE AND ORIENT MODULE ARCHITECTURE PHASE BOXED IN GREEN WITHIN THE NASA PROJECT LIFE-CYCLE BASED ON NASA PROCEDURAL REQUIREMENTS NPR 7123.1B [78].

The scope of the architecture was limited for this research to fit within the available project resource constraints (i.e. workforce availability, schedule) and operate in an early project environment where many system elements are still ill-defined:

- Single reference architecture for a candidate system concept
- Module through Assembly levels on the Axis of Organization
- Primarily the hardware aspects of the robotic system (the software and avionics aspects of the architecture will be developed separately within CCRS)
- Interaction (type B) scenarios that focus primarily on the direct interactions between the system of interest and the actors and external systems within the system context for the top-level COM scenarios [79]
- Primary portion of the COM main scenario that starts from OS capture and ends with OS assembly into the PCV (development of commissioning, decommissioning, and support services will be developed at later phases of the project life-cycle)

This architecture effort addresses architectural concerns from all domains within the Axis of Categories and captures the associated knowledge within the architecture description. Examples of the types of knowledge that the architecting team intended to capture for each domain include:

- Robotics: links, stroke, preload, alignment accuracy, optical interfaces, fiducials
- Mechanical: structural components, mechanical interfaces, mass, mechanism elements
- Electrical: sensors, motor control interfaces, power, harness, data rate, time durations
- Thermal: PRT interfaces, heater power interfaces, thermal control system elements
- Flight Software: data, control flow, activities, decision nodes, mission threads
- Contamination Control/Planetary Protection: Planetary Protection requirements, materials, seals, capture approach

4.2.5. Research Purpose

The purpose of this research is to quantitatively measure how completely an MBSE approach captures architectural knowledge relative to a non-MBSE approach when applied to architecting a robotic space system. To accomplish this, architecture descriptions for a robotic space system will be generated in parallel following both an MBSE and a non-MBSE approach to perform a side-by-side comparative analysis of the two approaches. The completeness of architectural knowledge capture will be measured based on the quantity of architectural knowledge elements that properly align with the knowledge constructs utilized by each approach along various knowledge categories. The MSR CCRS Capture and Orient Module will be used as a representative robotic space system with high complexity as a case study. This research aims to contribute to the MBSE literature through providing:

- Quantitative evidence of MBSE's advantages over traditional, non-MBSE approaches
- A case study that compares an MBSE approach side-by-side with a non-MBSE approach

- A case study of MBSE and SysML applied within the robotics applications domain
- A methodology to compare MBSE and non-MBSE approaches through knowledge categorization

4.3. Methodology

This research effort carried out the steps depicted in Figure 11 to compare the accuracy of the architecture knowledge content formally captured by an MBSE approach vs. non-MBSE approach for describing a robotic space system architecture. The architecture development took place over the course of two years by the COM engineering team at the Jet Propulsion Lab (JPL) in Pasadena, California. Both the MBSE and non-MBSE approaches were carried out simultaneously by the same engineering team during the two-year duration.

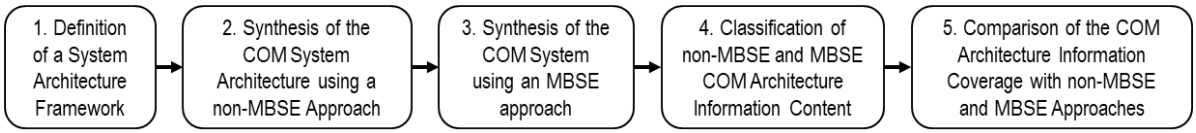


FIGURE 11. STEPS CARRIED OUT TO COMPARE THE ACCURACY OF THE ARCHITECTURE KNOWLEDGE CONTENT FORMALLY CAPTURED BY AN MBSE APPROACH VS. NON-MBSE APPROACH FOR DESCRIBING A ROBOTIC SPACE SYSTEM ARCHITECTURE.

4.3.1. Step 1: Definition of a System Architecture Framework

A framework was developed to capture the architecture information content, guide the architecting process, and provide views of the system from different perspectives (see Figure 12). The framework was synthesized from frameworks defined in the MBSAP [30], MagicGrid [44], and STRATA [41] methods. A table format, similar to that used by MagicGrid, was adopted due to its visual representation, which aides in communicating, understanding, and tracking the architecting process. The axis of organization from the system architecture taxonomy (shown in Figure 9) was chosen for the rows of the table. The structure, data,

behavior, and requirements perspectives from MBSAP were chosen for the columns of the table. The system architecture information content (e.g., system elements, element properties, and relationships to describe the architecture) that was captured within each level in each perspective column is also listed in Figure 12. This content was selected based on inputs from the CCRS architecture team at JPL and guidance from the NASA Systems Engineering Handbook [77], Expanded Guidance for NASA Systems Engineering [111], and MBSAP [30].

		Perspective			
		Structure	**Data**	**Behavior**	**Requirements**
Organization Level	**Module Level (L4)**	• **Glossary:** Terms, Term Descriptions • **Classifications:** Hierarchy Levels, Interface Types, Element Ownerships • **Product Breakdown:** Product Breakdown Structure, Element Types, Element Hierarchy Levels, Element Ownership Assignments, Structural Decompositions • **Block Diagrams:** System Block Diagram, Parts, Part Types, Part Ownerships, Part Ownership Assignments, Interfaces, Interface Type Assignments, Flow Item Allocations, Flow Directions • **Specifications:** Description, Owner, Attributes, Allocated Functions, Data Generation and Use, External Interfaces, Allocated Requirements, Applicable Design Principles, Recommended Approaches, Approach Rationales, Risks, Development Strategies, Development Strategy Rationales, Core Competencies	• **Glossary:** Terms, Term Descriptions • **Classifications:** Hierarchy Levels • **Data Model:** Hierarchical Data Model, Data Elements, Data Element Hierarchy Levels, Specializations	• **Glossary:** Terms, Term Descriptions • **Scenarios:** Functional Flows, Functional Decompositions, Functions, Function Hierarchy Levels, Control Flows, Functional Allocations	• **Glossary:** Terms, Term Descriptions • **Classifications:** Hierarchy Levels, Requirement Types • **Requirements:** Requirements Hierarchy, Requirements, Requirement IDs, Requirement Hierarchy Levels, Requirement Requirement Types, Requirement Texts, Requirement Rationales, Requirement Types, Verification Methods, Parents, Requirements Allocations
	Subsystem Level (L5)				
	Assembly Level (L6)				

FIGURE 12. ARCHITECTURE FRAMEWORK USED FOR DEVELOPING THE COM ARCHITECTURE WITH A LISTING OF THE SYSTEM INFORMATION CONTENT FOR ALL ORGANIZATION LEVELS WITHIN EACH COLUMN.

4.3.2. Step 2: Synthesis of the COM System Architecture Description using a non-MBSE Approach

A description of the COM architecture was synthesized using non-MBSE tools (traditional, document-centric systems engineering tools used to generate spreadsheets, presentations, and manuscripts) within the architecture framework established in Step 1. For the non-MBSE approach, Microsoft PowerPoint, Word, and Excel were utilized to implement the non-MBSE tasks due to their compatibility with commonly produced document-based systems engineering artifacts used at JPL in Pre-Phase A, as well as familiarity with the engineering team for

capturing and communicating numerical, textual, and graphical system information. Although many deeper functionalities are available within these tools, this project sought to represent the function of these non-MBSE tools in the ways that they are commonly exercised within JPL's robotics space systems engineering teams.

A high-level information model was developed from the architecture framework in Figure 12 to lay out the non-MBSE document artifacts to cover the architecture information content for each perspective and organization level (see Figure 13). For the Product Breakdown, Data Model, Scenario, and Requirements, single documents cover all three levels of organization. For the Block Diagrams and Specifications, independent documents cover each level of organization. The full set of documents developed for the COM are shown in Figure 14 and organized within the architecture framework developed in Figure 12. The non-MBSE artifacts generated include:

- COM Glossary: Excel spreadsheet containing a list of terms used in the COM
- COM Product Breakdown: PowerPoint slide showing the COM product decomposition from the Module Level (Level 4) down to the Assembly Level (Level 6)
- Block diagram slides: PowerPoint slides for the COM, COM subsystems, and COM assemblies. They display the individual elements of each assembly, along with heater power, PRT, separation device, servo motor control, workhorse motor control, optical, data, sensor power, mechanical, and temporary interfaces
- Specification manuscripts: Word documents for the COM, COM subsystems, and COM assemblies. They capture a textual description of the system element, its key attributes, the functions it performs, the requirements it must meet, and other relevant technical and programmatic characteristics

- COM Data Model: PowerPoint slide showing the COM data elements specialized for each system level
- COM Scenario: Excel spreadsheet listing the sequence of steps the system carries out broken down by system level. The main operational behavior of the system is captured through an operational scenario that represents the most common sequence of steps used to satisfy the system requirements
- COM Requirements: Excel spreadsheet listing level 4 through 6 requirements and associated requirements attributes

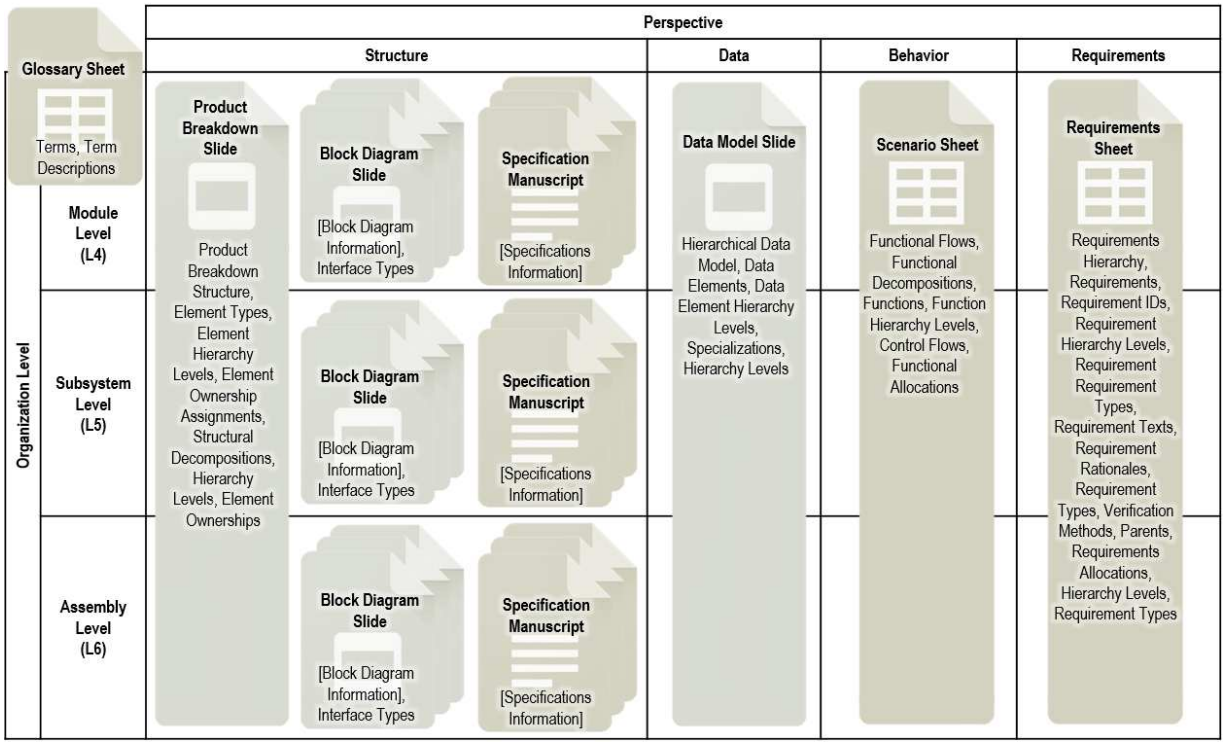


FIGURE 13. INFORMATION MODEL FOR THE NON-MBSE APPROACH SHOWING INDIVIDUAL DOCUMENT TYPES, THE INFORMATION CONTENT CAPTURED IN EACH TYPE OF DOCUMENT, AND THE TYPES OF DOCUMENTS GENERATED IN EACH ARCHITECTURE ORGANIZATION LEVEL AND PERSPECTIVE.

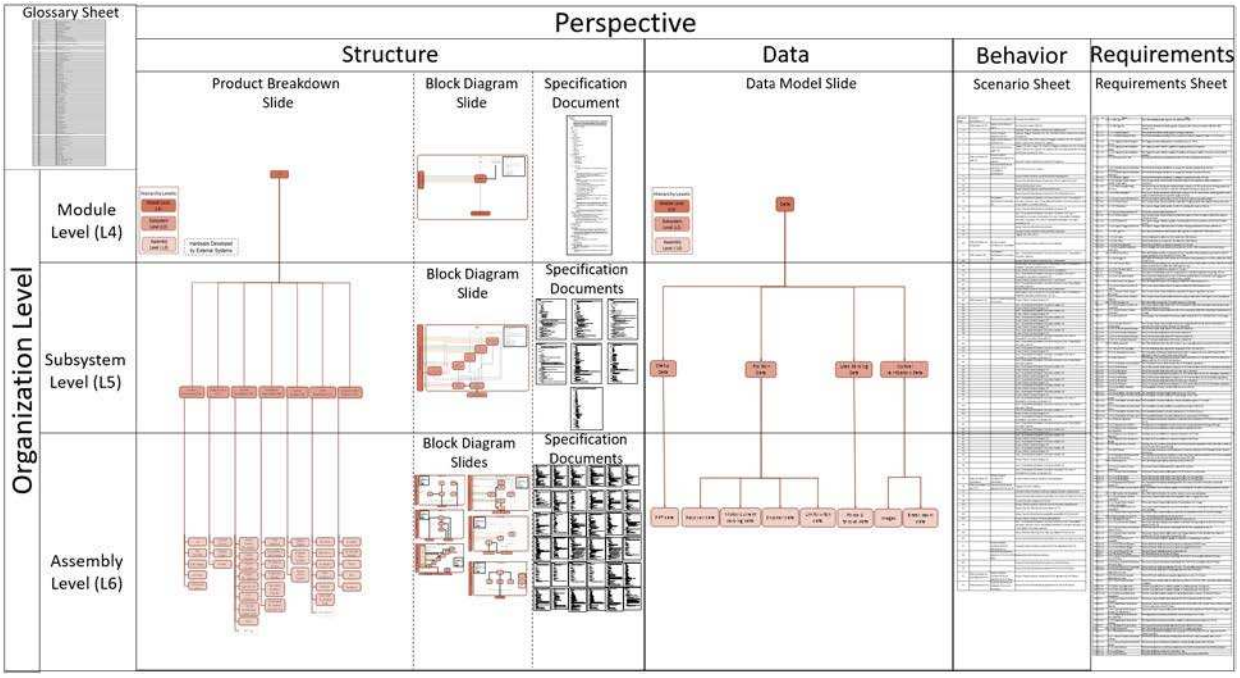


FIGURE 14. FULL VIEW OF CAPTURE AND ORIENT MODULE NON-MBSE ARCHITECTURE DESCRIPTION.

The process followed to architect the system from Level 4 (definition of the COM module and its relationships to other modules within the CCRS system) down to Level 6 (definition of the structural elements of the individual assembly and their relationships) is shown in the flowchart in Figure 15. During each step of the process, information elements were created for each perspective at each system level, and then added to the appropriate documents specified in Figure 13. With the document-based systems engineering approach, information elements were captured and distributed amongst the separate, independent documents (rather than a primary, integrated system model as in the MBSE approach). The document entries, diagrams, and tables were generated following typical systems engineering document formats, such as product breakdown structure diagrams shown in the NASA Systems Engineering Handbook [77], block diagrams shown in Spacecraft Systems Engineering [112], and scenario templates shown in Requirements Engineering [79].

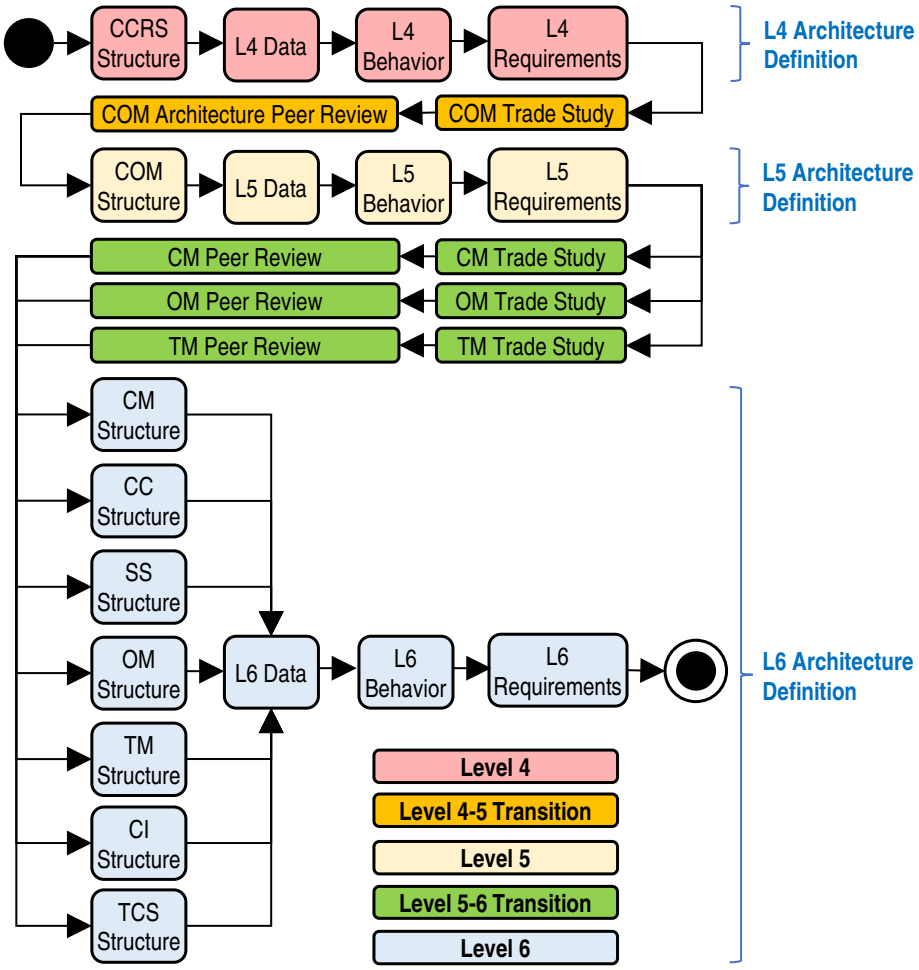


FIGURE 15. FLOWCHART OF THE COM ARCHITECTING PROCESS IMPLEMENTED DOWN TO LEVEL 6.

4.3.3. Step 3: Synthesis of the COM system Architecture Description using an MBSE Approach

A description of the COM architecture was synthesized using an MBSE tool within the architecture framework established in Step 1. For the MBSE approach, Cameo Systems Modeler was chosen as the architecting software tool, and SysML was chosen as the architecting language per guidance by the MSR Campaign. At the conception of the project, JPL and European Space Agency (ESA) established an overall MSR campaign model to integrate technical and

programmatic information across all missions and mission elements [113]. SysML and MagicDraw was chosen by the campaign as the implementing modelling language and tool due to the extensive experience and resources of both agencies. The MSR campaign model provided initial definition of all mission phases and associated operational scenarios were defined, along with a model hierarchy to link the various SRL and ERO project models [114]. The MSR campaign model also provided definition of the first three levels of the MSR architecture: Level 1 (MSR Campaign), Level 2 (ERO Mission), L3 (CCRS Payload). These served as the initial inputs and basis for the COM system architecture description. The COM architecture model was implemented in Cameo Systems Modeler (the rebrand of MagicDraw) using the SysML language profile to be compatible with and allow integration into the MSR campaign model within the Teamwork Cloud infrastructure set up on the JPL server.

The modeling method used in the MBSE approach was designed to follow the NASA Systems Engineering Engine System Design Process [78] and built upon elements from MBSAP [30] and STRATA [41] methodologies. The NASA System Design process defines system requirements and technical solutions for each system level in a recursive manner, starting at the top product level, and then progressing down to the lowest level necessary to converge the system design. The modeling method incorporated a general flow that followed the MBSAP methodology, where top level of the system is defined in an Operational Viewpoint, the lower levels in the Logical/Functional Viewpoint, and final level of the design in the Physical Viewpoint [30]. A layered approach to define the architecture following the STRATA methodology [41] and NASA Systems Engineering Engine System Design Process [78] was then taken, where each level of the architecture was defined prior to drilling down to the next lower, more specific levels.

A high-level information model was developed from the architecture framework in Figure 12 to lay out the MBSE views to cover the architecture information content for each perspective and organizational level (see Figure 16). For the Product Breakdown, Data Model, and Requirements, single diagrams cover all three levels of organization. For the Internal Block Diagrams and Activity Diagrams, independent diagrams cover each level of organization. The full set of diagrams generated with the CCRS COM model using Cameo Systems Modeler are shown in Figure 17 and organized within the architecture framework developed in Figure 12. The MBSE artifacts generated include:

- COM Glossary: Glossary table containing a list of terms used in the COM
- COM Product Breakdown: SysML block definition diagram showing the COM product decomposition from the Module Level (Level 4) down to the Assembly Level (Level 6). Blocks specifications capture and display a textual description of the system element, its key attributes, the functions it performs, the requirements it must meet, and other relevant technical and programmatic characteristics
- Block diagrams: SysML internal block diagrams for the COM, COM subsystems, and COM assemblies. They display the individual elements of each assembly, along with heater power, PRT, separation device, servo motor control, workhorse motor control, optical, data, sensor power, mechanical, and temporary interfaces.
- COM Data Model: SysML block definition diagram showing the COM data elements specialized for each system level
- Activity diagrams: SysML activity diagrams showings sequences of actions carried out at each system level

- COM Requirement Hierarchy: SysML requirements diagram showing the COM requirements decomposition from the Module Level (Level 4) down to the Assembly Level (Level 6). Requirements specifications capture and display associated requirements attributes

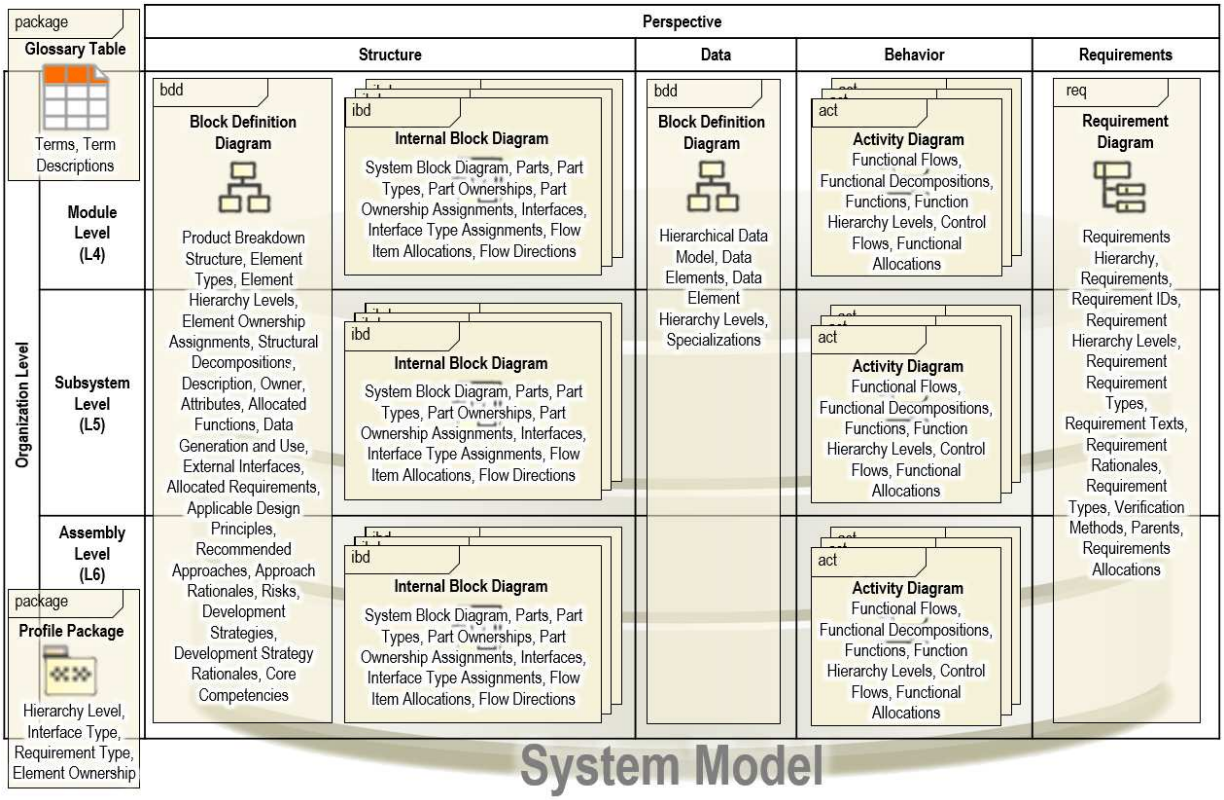


FIGURE 16. INFORMATION MODEL FOR THE SYSML-BASED MBSE APPROACH SHOWING INDIVIDUAL DIAGRAM TYPES, THE INFORMATION CONTENT CAPTURED IN EACH TYPE OF DIAGRAM, AND THE TYPES OF DIAGRAMS GENERATED IN EACH ARCHITECTURE ORGANIZATION LEVEL AND PERSPECTIVE. THE SYSTEM MODE.

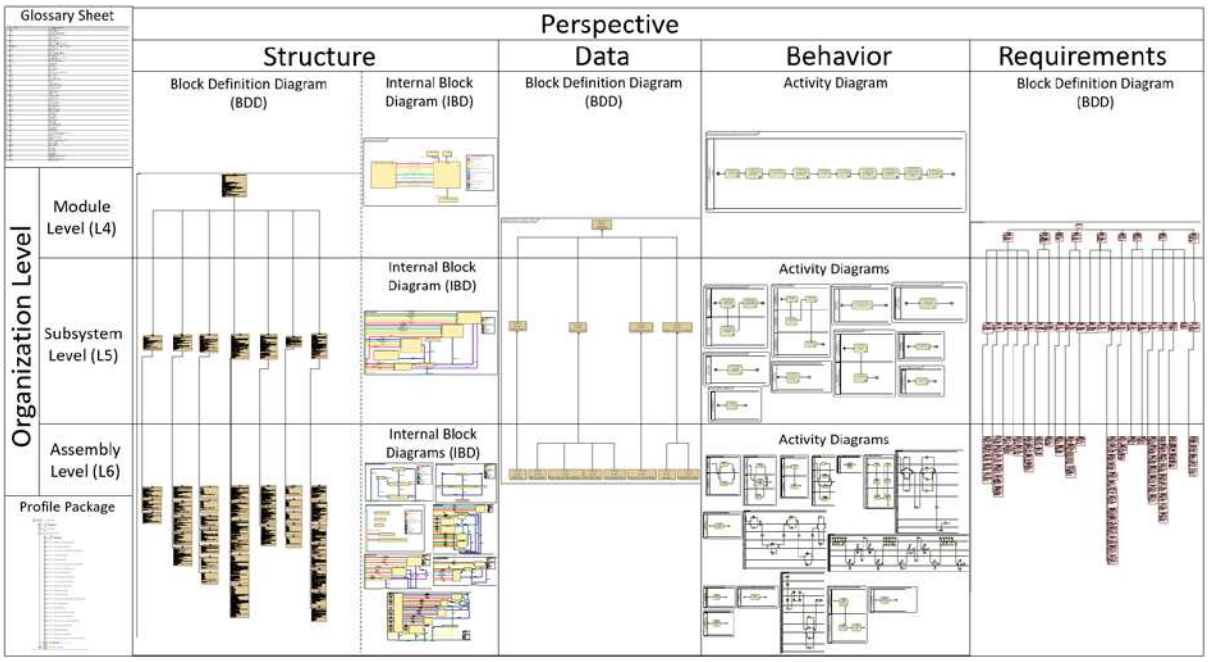


FIGURE 17. FULL VIEW OF CAPTURE AND ORIENT MODULE MBSE ARCHITECTURE DESCRIPTION.

Similar to the non-MBSE approach, the MBSE approach followed the same general process shown in Figure 15. During each step of the process, information elements were generated for each perspective at each system level, integrated into a system model using Cameo System Modeler, and referenced in the appropriate model views specified in Figure 16. The model elements, diagrams, and views were generated following standard SysML diagram formats, such as those shown in Effective Model-Based Systems Engineering [30], and implemented following standard Cameo System Modeler modeling procedures, such as those outlined in the Cameo Systems Modeler User Guide book [29].

4.3.4. Step 4: Classification of non-MBSE and MBSE COM Architecture Information Content

A knowledge taxonomy was chosen to classify the COM architecture information content, and then classified in a table based on its knowledge category. Following the classification of

the architecture information content, the tool constructs used to represent the information content within each of the MBSE and non-MBSE tools were categorized based on how the tool constructs captured and stored the information. The knowledge dimension of the Revised Bloom's Taxonomy [102] was chosen as a means for classifying each architecture's information content. Bloom's Taxonomy has its basis in modern cognitive science and cognitive psychology perspectives on knowledge representation, and has a detailed knowledge classification structure that encompasses both concrete and abstract categories of knowledge. This means of classifying information content from the set of systems architecting artifacts recognizes that systems engineering is a human-centered discipline [33], requiring teaming, leadership, management, design, and learning to enable complex systems over the system life cycle [115]. This study therefore asserts that system architecting knowledge consists of factual, conceptual, and procedural knowledge of the system architecture, as well as collections of principles, heuristics, and methods that constitute meta-cognitive knowledge of the system architecture.

The knowledge dimension of the Revised Bloom's Taxonomy organizes knowledge into four categories: factual, conceptual, procedural, and metacognitive. Factual knowledge (Category A) consists of discrete, isolated content elements and terms. Conceptual knowledge (Category B) consists of interrelationships between basic content elements to form larger, more organized, abstract bodies of knowledge, such as categories, principles, and structures. Procedural knowledge (Category C) consists of knowledge of how to perform a task, such as algorithms, methods, and conditional knowledge required for selecting a method or procedure. Metacognitive knowledge (Category D) consists of higher order knowledge regarding control over cognitive tasks, strategies, and self-knowledge relative to specific subject matters. The four categories of knowledge are further divided into eleven subcategories, as shown in Table 1.

Revised Bloom's Taxonomy knowledge dimension categories, subcategories, and classified system knowledge types.. Within the context of system architecting, factual knowledge is associated with the physical architecture and its design details, conceptual knowledge is associated with the allocated architecture and its high-level design structure, procedural knowledge is associated with the architecture methods and behaviors, and metacognitive knowledge is associated with the system architecting process itself.

TABLE 1. REVISED BLOOM'S TAXONOMY KNOWLEDGE DIMENSION CATEGORIES, SUBCATEGORIES, AND CLASSIFIED SYSTEM KNOWLEDGE TYPES.

	Category		Subcategory		Classified Knowledge
Knowledge	A	Factual	A.A	Terminology	Term
			A.B	Specific Details/Elements	Allocated Requirement, Attribute, Data Element Hierarchy Level, Data Generation and Use, Description, Element Hierarchy Level, Element Ownership Assignment, External Interface, Flow Direction, Flow Item Allocation, Function, Function Hiearchy Level, Interface, Interface Type Assignment, Owner, Parent, Part, Part Ownership Assignment, Part Type, Reference Function, Requirement, Requirement Allocation, Requirement Hierarchy Level, Requirement ID, Requirement Rationale, Requirement Requirement Type, Requirement Text, Term Description
	B	Conceptual	B.A	Classifications/Categories	Data Element, Element Ownership, Element Type, Hierarchy Level, Interface Type, Part Ownership, Requirement Type
			B.B	Principles/Generalizations	Applicable Design Principle, Specialization
			B.C	Theories/Models/Structures	Allocated Function, Functional Allocation, Functional Decomposition, Hierarchical Data Model, Product Breakdown Structure, Requirements Hierarchy, Structural Decomposition, System Block Diagram
	C	Procedural	C.A	Subject-specific Skills/ Algorithms	Control Flow, Functional Flow
			C.B	Subject-specific Techniques/ Methods	Recommended Approach
			C.C	Criteria for Procedure Use	Approach Rationale
	D	Metacognitive	D.A	Strategies	Development Strategy, Verification Method
			D.B	Cognitive Tasks	Development Strategy Rationale, Risk
			D.C	Self-knowledge	Core Competency

To classify the non-MBSE and MBSE COM architecture information content, the types of information content from Figure 12. Architecture framework used for developing the COM architecture with a listing of the system information content for all organization levels within each column. was first mapped to the Revised Bloom's Taxonomy knowledge categories. To perform the mapping, information types were categorized based on the knowledge category definitions defined in the Revised Bloom's Taxonomy book by Anderson et al. [102], along with

personal discussions with Richard Mayer, a cognitive psychologist at the University of California Santa Barbara and co-author of the Revised Bloom's Taxonomy. Table 1 shows the system architecture information content listed in Figure 12. Architecture framework used for developing the COM architecture with a listing of the system information content for all organization levels within each column. mapped to the Revised Bloom's Taxonomy knowledge categories and subcategories. Next, the tool constructs used to represent the information content within each of the MBSE and non-MBSE tools was categorized based on how the tool construct capture and stores the information. Table 2 shows how each of the MBSE and non-MBSE tools were used to represent each of the knowledge item types, which of the Revised Bloom's Taxonomy knowledge categories that tool representation captures, and whether or not it aligns with the actual knowledge category of the system knowledge type.

TABLE 2. SYSTEM KNOWLEDGE TYPES CAPTURED IN MBSE AND NON-MBSE TOOLS CLASSIFIED USING THE REVISED BLOOM'S TAXONOMY KNOWLEDGE DIMENSION. GREEN INDICATES KNOWLEDGE CLASSIFICATION ALIGNMENT WITH THE SYSTEM KNOWLEDGE CLASSIFICATION, WHILE RED INDICATES NON-ALIGNMENT

System Knowledge	Knowledge Classification	MBSE Tool	MBSE Representation	MBSE Representation Knowledge Classification	Non-MBSE Tool	Non-MBSE Representation	Non-MBSE Representation Knowledge Classification
Allocated Function	B.C. Theories/	Cameo	Allocated From	B.C. Theories/ Models/ Structures	Word	Text	A.B. Specific Details/ Elements
Allocated Requirement	A.B. Specific Details/	Cameo	Satisfies	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Applicable Design Principle	B.B. Principles/	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Approach Rationale	C.C. Criteria for	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Attribute	A.B. Specific Details/	Cameo	Values	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Control Flow	C.A. Subject-specific	Cameo	Control Flow	C.A. Subject-specific Skills/	Excel	Cell Location	A.B. Specific Details/ Elements
Core Compentency	D.C. Self-knowledge	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Data Element	B.A. Classifications/	Cameo	Block	B.A. Classifications/ Categories	PowerPoint	Text Box	A.B. Specific Details/ Elements
Data Element Hierarchy Level	A.B. Specific Details/	Cameo	Stereotype	A.B. Specific Details/ Elements	PowerPoint	Text Box Color	A.B. Specific Details/ Elements
Data Generation and Use	A.B. Specific Details/	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Description	A.B. Specific Details/	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Development Strategy	D.A. Strategies	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Development Strategy Rationale	D.B. Cognitive Tasks	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Element Hierarchy Level	A.B. Specific Details/	Cameo	Stereotype	A.B. Specific Details/ Elements	PowerPoint	Text Box Color	A.B. Specific Details/ Elements
Element Ownership	B.A. Classifications/	Cameo	Stereotype	B.A. Classifications/ Categories	PowerPoint	Text Box Style	A.B. Specific Details/ Elements
Element Ownership Assignment	A.B. Specific Details/	Cameo	Block Type	A.B. Specific Details/ Elements	PowerPoint	Text Box Style	A.B. Specific Details/ Elements
Element Type	B.A. Classifications/	Cameo	Block	B.A. Classifications/ Categories	PowerPoint	Multilevel List	B.A. Classifications/ Categories
External Interface	A.B. Specific Details/	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Flow Direction	A.B. Specific Details/	Cameo	Item Flow Direction	A.B. Specific Details/ Elements	PowerPoint	Arrow	A.B. Specific Details/ Elements
Flow Item Allocation	A.B. Specific Details/	Cameo	Item Flow	A.B. Specific Details/ Elements	PowerPoint	Text Box	A.B. Specific Details/ Elements
Function	A.B. Specific Details/	Cameo	Call Behavior Action	A.B. Specific Details/ Elements	Excel	Text	A.B. Specific Details/ Elements
Function Hierarchy Level	A.B. Specific Details/	Cameo	Applied Stereotype	A.B. Specific Details/ Elements	Excel	Column	A.B. Specific Details/ Elements
Functional Allocation	B.C. Theories/	Cameo	Allocated To	B.C. Theories/ Models/ Structures	Excel	Text	A.B. Specific Details/ Elements
Functional Decomposition	B.C. Theories/	Cameo	Behavior	B.C. Theories/ Models/ Structures	Excel	Cell Location	A.B. Specific Details/ Elements
Functional Flow	C.A. Subject-specific	Cameo	Diagram	C.A. Subject-specific Skills/	Excel	Cell Sequence	C.A. Subject-specific Skills/
Hierarchical Data Model	B.C. Theories/	Cameo	SysML BDD	B.C. Theories/ Models/ Structures	PowerPoint	Slide	B.C. Theories/ Models/ Structures
Hierarchy Level	B.A. Classifications/	Cameo	Stereotype	B.A. Classifications/ Categories	PowerPoint	Text Box Color	A.B. Specific Details/ Elements
Interface	A.B. Specific Details/	Cameo	Connector	A.B. Specific Details/ Elements	PowerPoint	Line	A.B. Specific Details/ Elements
Interface Type	B.A. Classifications/	Cameo	Stereotype	B.A. Classifications/ Categories	PowerPoint	Line Format	A.B. Specific Details/ Elements
Interface Type Assignment	A.B. Specific Details/	Cameo	Connector Type	A.B. Specific Details/ Elements	PowerPoint	Line Format	A.B. Specific Details/ Elements
Owner	A.B. Specific Details/	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Parent	A.B. Specific Details/	Cameo	Derived From	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Part	A.B. Specific Details/	Cameo	Part Property	A.B. Specific Details/ Elements	PowerPoint	Text Box	A.B. Specific Details/ Elements
Part Ownership	B.A. Classifications/	Cameo	Block Type	B.A. Classifications/ Categories	PowerPoint	Text Box Text	A.B. Specific Details/ Elements
Part Ownership Assignment	A.B. Specific Details/	Cameo	Part Property Type	A.B. Specific Details/ Elements	PowerPoint	Text Box Style	A.B. Specific Details/ Elements
Part Type	A.B. Specific Details/	Cameo	Block Type	A.B. Specific Details/ Elements	PowerPoint	Text Box Text	A.B. Specific Details/ Elements
Product Breakdown Structure	B.C. Theories/	Cameo	SysML BDD	B.C. Theories/ Models/ Structures	PowerPoint	Slide	B.C. Theories/ Models/ Structures
Recommended Approach	C.B. Subject-specific	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Reference Function	A.B. Specific Details/	Cameo	Traced To	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Requirement	A.B. Specific Details/	Cameo	Requirement Block	A.B. Specific Details/ Elements	Excel	Row	A.B. Specific Details/ Elements
Requirement Allocation	A.B. Specific Details/	Cameo	Satisfied By	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Requirement Hierarchy Level	A.B. Specific Details/	Cameo	Applied Stereotype	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Requirement ID	A.B. Specific Details/	Cameo	ID	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Requirement Rationale	A.B. Specific Details/	Cameo	Documentation	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Requirement Requirment Type	A.B. Specific Details/	Cameo	Applied Stereotype	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Requirement Text	A.B. Specific Details/	Cameo	Text	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Requirement Type	B.A. Classifications/	Cameo	Stereotype	B.A. Classifications/ Categories	Excel	Cell Text	A.B. Specific Details/ Elements
Requirements Hierarchy	B.C. Theories/	Cameo	SysML BDD	B.C. Theories/ Models/ Structures	Excel	Cell Text	A.B. Specific Details/ Elements
Risk	D.B. Cognitive Tasks	Cameo	Documentation	A.B. Specific Details/ Elements	Word	Text	A.B. Specific Details/ Elements
Specialization	B.B. Principles/	Cameo	Generalization	B.B. Principles/ Generalizations	PowerPoint	Arrow	A.B. Specific Details/ Elements
Structural Decomposition	B.C. Theories/	Cameo	Composition	B.C. Theories/ Models/ Structures	PowerPoint	Arrow	A.B. Specific Details/ Elements
System Block Diagram	B.C. Theories/	Cameo	SysML IBD	B.C. Theories/ Models/ Structures	PowerPoint	Slide	B.C. Theories/ Models/ Structures
Term	A.A. Terminology	Cameo	Term	A.A. Terminology	Excel	Row	A.A. Terminology
Term Description	A.B. Specific Details/	Cameo	Description	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements
Verification Method	D.A. Strategies	Cameo	Verification Method	A.B. Specific Details/ Elements	Excel	Cell Text	A.B. Specific Details/ Elements

4.3.5. Step 5: Comparison of the COM Architecture Information Coverage with non-MBSE and MBSE Approaches

For both the non-MBSE documents and MBSE approaches, the number of information content items and their categorical alignment within the knowledge taxonomy (based on the

classifications shown in Table 2) were counted. The analysis was performed through carrying out the following sub-steps:

- Step 5.1: A spreadsheet was generated listing each general information type (from the System Knowledge listed in Table 2) that was captured in both the MBSE and non-MBSE artifacts within glossary, classifications, product breakdown, block diagrams, specifications, data model, scenarios, and requirements artifacts
- Step 5.2: The quantities of specific information items defined within the architecture for each information type were manually counted and entered into the table. These information items were identified by manually analyzing each document entry (for the non-MBSE documents) and each model element (for the MBSE model)
- Step 5.3: The quantities of information items whose representations align with the actual knowledge classification of the system knowledge item were entered into the table for both MBSE and non-MBSE approaches based on the alignment and non-alignment classifications determined in Step 4 and shown in Table 2
- Step 5.4: The number of aligned information items were summed in the spreadsheet for both the MBSE and non-MBSE approaches and compared

An example of how this task was done for knowledge pertaining to the "Transfer Mechanism Transfer OS to Orientation Mechanism" activity is shown in Figure 18. MBSE SysML activity diagram representation of "Transfer Mechanism Transfers OS to Orientation Mechanism" activity., Table 3, and Table 4. Figure 18. MBSE SysML activity diagram representation of "Transfer Mechanism Transfers OS to Orientation Mechanism" activity. shows the MBSE representation of the activity using a SysML activity diagram produced in Cameo System Modeler. Table 3 shows the non-MBSE representation of the activity within a spreadsheet

produced in Microsoft Excel. Table 4 shows the information content items that define the activity, the constructs used to represent the knowledge within each of the MBSE and non-MBSE tools, how these constructs represent the knowledge in terms of knowledge category, and whether these representations align with the actual knowledge classification of the system knowledge element. In this example, the MBSE approach more richly represented the system knowledge than the non-MBSE approach, as measured by the number and types of information content present in the artifact. This is due to the fact that the MBSE activity diagram contains the modeling constructs to explicitly represent functional decomposition, allocation of functions to system elements, and control flows, while the non-MBSE spreadsheet does not contain the constructs for these knowledge elements. For the non-MBSE spreadsheet, these items must be inferred through interpreting written text within the cells and soft guidance from the locations of the cells relative to one another in the spreadsheet.

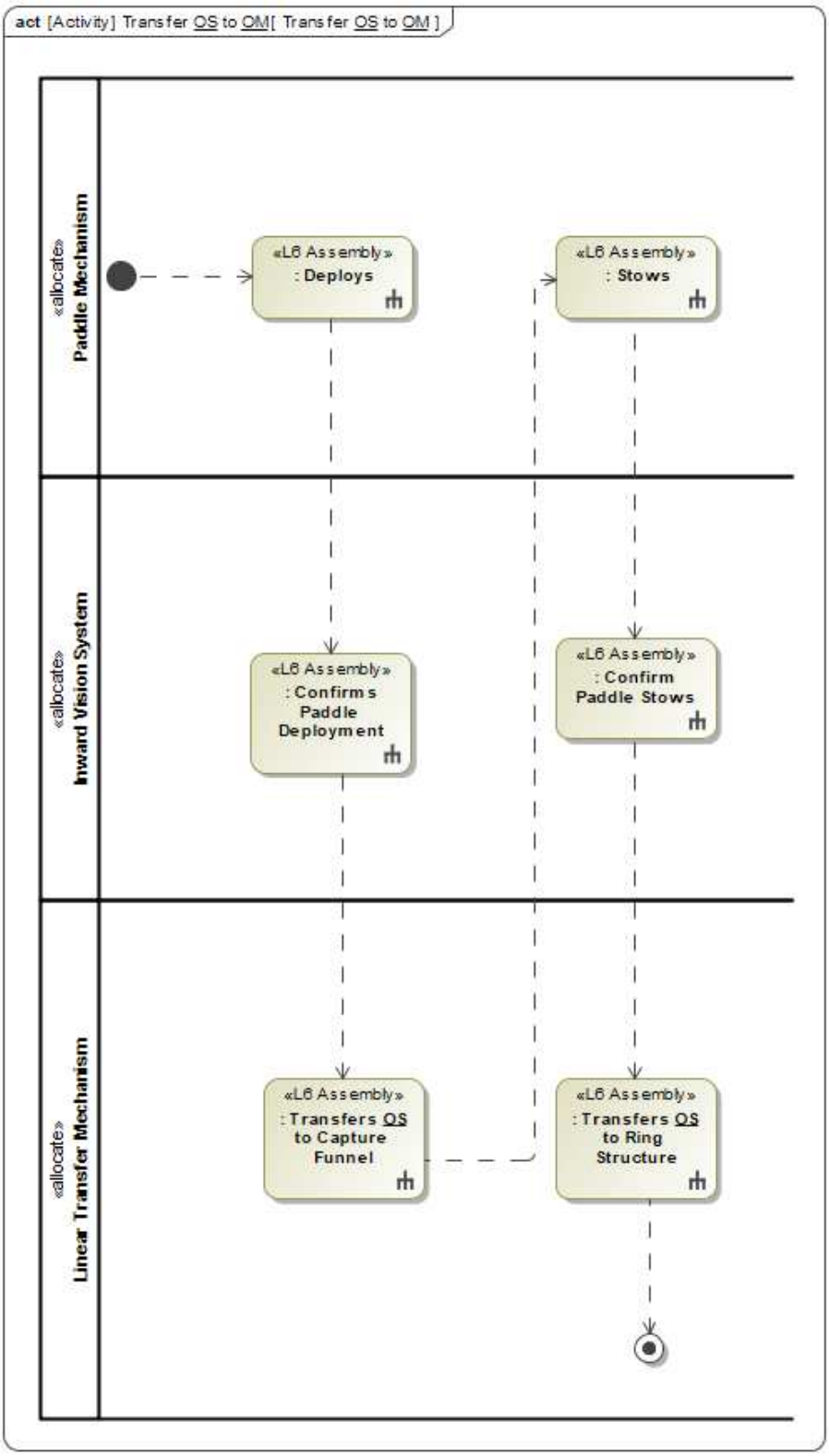


FIGURE 18. MBSE SYSML ACTIVITY DIAGRAM REPRESENTATION OF "TRANSFER MECHANISM TRANSFERS OS TO ORIENTATION MECHANISM" ACTIVITY.

TABLE 3. NON-MBSE SPREADSHEET DOCUMENT TEXTUAL REPRESENTATION OF "TRANSFER MECHANISM TRANSFERS OS TO ORIENTATION MECHANISM" ACTIVITY.

Scenario Description L5	Scenario Description L6
Transfer Mechanism transfers OS to Orientation Mechanism	Paddle Mechanism deploys
	Inward Vision System confirms Paddle deployment
	Linear Transfer Mechanism transfers OS to Capture Funnel
	Paddle Mechanism stows
	Inward Vision System confirms Paddle stow
	Linear Transfer Mechanism transfers OS to Ring Structure

TABLE 4. KNOWLEDGE CLASSIFICATION OF "TRANSFER MECHANISM TRANSFERS OS TO ORIENTATION MECHANISM" ACTIVITY. GREEN INDICATES KNOWLEDGE CLASSIFICATION ALIGNMENT WITH THE SYSTEM KNOWLEDGE CLASSIFICATION, WHILE RED INDICATES NON-ALIGNMENT.

System Knowledge	Knowledge Classification	Knowledge Quantity	MBSE Knowledge Representation	MBSE Knowledge Classification	MBSE Quantity Aligned	Non-MBSE Knowledge Representation	Non-MBSE Knowledge Classification	Non-MBSE Quantity Aligned
Function	A.B. Elements	6	Call Behavior Action	A.B. Elements	6	Text	A.B. Elements	6
Function Hierarchy Level	A.B. Specific Details	6	Call Behavior Action Applied Stereotype	A.B. Specific Details	6	Column	A.B. Specific Details	6
Functional Decomposition	B.C. Structures	6	Call Behavior Action Behavior	B.C. Structures	6	Cell Location	A.B. Specific Details	0
Functional Allocation	B.C. Models	6	Allocated To	B.C. Models	6	Text	A.B. Specific Details	0
Functional Flow	C.A. Algorithms	1	SysML Activity Diagram	C.A. Algorithms	1	Cell Sequence	C.A. Algorithms	1
Control Flow	C.A. Algorithms	7	Control Flow	C.A. Algorithms	7	Cell Location	A.B. Specific Details	0
Total Elements		32	Total MBSE Aligned		32	Total Non-MBSE Aligned		13

This type of analysis was repeated for all the system knowledge content captured within the full set of non-MBSE documents in Figure 14. Full view of Capture and Orient Module non-MBSE architecture description. and MBSE diagrams in Figure 17. Full view of Capture and Orient Module MBSE architecture description.. The results of these analyses were used to compare how the non-MBSE and MBSE approaches captured information content for the COM architecture. The full tally of system elements, categorized by knowledge category, along with

the number of MBSE and non-MBSE elements that aligned with the knowledge category, is shown in Table 5. A total of 4,389 knowledge elements were analyzed and compared.

TABLE 5. TOTAL NUMBER OF COM SYSTEM ARCHITECTURE KNOWLEDGE ELEMENTS AND NUMBER OF ALIGNMENTS FOR BOTH MBSE AND NON-MBSE APPROACHES ORGANIZED BY KNOWLEDGE CATEGORY. GREEN INDICATES A HIGH NUMBER OF KNOWLEDGE ELEMENT ALIGNMENTS, YELLOW INDICATES A MEDIUM NUMBER OF KNOWLEDGE ELEMENT ALIGNMENTS, AND RED INDICATES A LOW NUMBER OF KNOWLEDGE ELEMENT ALIGNMENTS RELATIVE TO THE TOTAL NUMBER OF KNOWLEDGE ELEMENTS FOR EACH KNOWLEDGE SUBCATEGORY.

	Category		Subcategory		Total Elements	MBSE Aligned	Non-MBSE Aligned
Knowledge	A	Factual	A.A	Terminology	73	73	73
			A.B	Specific Details/Elements	2921	2921	2921
	B	Conceptual	B.A	Classifications/Categories	148	148	46
			B.B	Principles/Generalizations	102	13	0
			B.C	Theories/Models/Structures	469	469	9
	C	Procedural	C.A	Subject-specific Skills/Algorithms	255	255	27
			C.B	Subject-specific Techniques/Methods	38	0	0
			C.C	Criteria for Procedure Use	55	0	0
	D	Metacognitive	D.A	Strategies	184	0	0
			D.B	Cognitive Tasks	100	0	0
			D.C	Self-knowledge	44	0	0
				Total	4389	3879	3076

4.4. Results and Discussion

The results and discussion section describes the comparison between the COM MBSE and non-MBSE based architecture descriptions. These comparisons are presented on the basis of the quantity of information coverage present in each architecture representation, high-level knowledge capture, information coverage across the architecture framework, and the suitability of artifacts for information capture.

4.4.1. Quantity of Information Coverage

Based on the results presented in Table 5, the MBSE architecture artifacts generated using the MBSE approach applied to COM system in this research provided more architecture information coverage relative to the non-MBSE approach based on knowledge categorization alignment (the tool correctly represented the knowledge element based on its Bloom's Taxonomy knowledge classification). Out of the 4,389 knowledge elements that made up the COM architecture, the MBSE approach was able to represent 3,879 of them, while the non-MBSE approach was able to represent 3,076 of them. With a more complete representation of information provided by the MBSE approach, there is less risk for information loss or mistranslation. A majority of these benefits were seen in the high-level knowledge categories of conceptual and procedural knowledge, where the MBSE approach used provides a means to more accurately represent categorical information through the definition of blocks and stereotypes, abstract structural relationships through the definition of directed compositions, functional allocations to structural elements, and procedures through the definition of control flows.

4.4.2. High-level Knowledge Capture

Based on the results presented in Table 5, the MBSE architecture artifacts generated using the MBSE approach applied to COM system in this research captured more high-level knowledge relative to the non-MBSE approach. Both MBSE and non-MBSE approaches can sufficiently represent the low-level knowledge categories listed in Category A, such as terminology, physical elements, value attributes, design details, and specific actions within an activity. However, the MBSE approach provided more coverage than the non-MBSE approach in the higher-level knowledge categories listed in Categories B and C. For example, in Subcategory B.A. (Classifications/Categories), the MBSE approach showed alignment with 148 knowledge

elements, where the non-MBSE approach showed alignment with 46 knowledge elements. Similar results were observed for subcategories B.B., B.C., and C.A.

The MBSE approach provided a partial ability to represent conceptual knowledge from Category B.B., within which generalization relationships are fully represented, but design principles are only partially represented via text. The non-MBSE approach provided partial ability to represent categorical information from Category B.A. through Microsoft Word system element templates, structural concepts from Category B.C. through Microsoft PowerPoint block diagrams, and procedures from Category C.A. through Microsoft Excel scenario sequences documented within columns. Both MBSE and non-MBSE architecture representations did not explicitly represent abstract procedural and metacognitive knowledge from categories C.B. through D.C. Instead, the information from these knowledge categories were primarily captured using text (categorized as Category A.B. Specific Details) in the block documentation fields within the MBSE approach, and as lines of text within the element specification documents for the non-MBSE approach.

4.4.3. Information Coverage Across the Architecture Framework

Based on the results presented in Table 6, the MBSE architecture artifacts generated using the MBSE approach applied to COM system in this research provided more architecture information coverage relative to the non-MBSE artifacts across more organization levels and perspectives within the architecture framework. For example, within the Structure Perspective at Level 6, the MBSE approach showed alignment with 1,898 knowledge elements, where the non-MBSE approach showed alignment with 1,423 knowledge elements. Similar results were observed for Levels 4 through 6 within the Structure, Data, and Behavior Perspectives.

TABLE 6. DISTRIBUTION OF COM SYSTEM KNOWLEDGE ELEMENTS AND NUMBER OF ALIGNMENTS FOR BOTH MBSE AND NON-MBSE APPROACHES SPECIFIC TO THE TWELVE ARCHITECTURE FRAMEWORK REGIONS. GREEN INDICATES A HIGH NUMBER OF KNOWLEDGE ELEMENT ALIGNMENTS, YELLOW INDICATES A MEDIUM NUMBER OF KNOWLEDGE ELEMENT ALIGNMENTS, AND RED INDICATES A LOW NUMBER OF KNOWLEDGE ELEMENT ALIGNMENTS RELATIVE TO THE TOTAL NUMBER OF KNOWLEDGE ELEMENTS FOR EACH REGION.

		Perspective											
		Structure			Data			Behavior			Requirements		
		Total	MBSE	Non-MBSE	Total	MBSE	Non-MBSE	Total	MBSE	Non-MBSE	Total	MBSE	Non-MBSE
Organization	Module Level (L4)	26	19	18	3	3	1	42	42	21	82	73	73
	Subystem Level (L5)	460	362	271	12	12	4	97	97	39	226	201	201
	Assembly Level (L6)	1898	1603	1423	24	24	8	644	644	236	709	633	633

A majority of the knowledge captured lied within the structural perspective at the assembly level (Level 6), followed by the other organization levels and perspectives. This shows the importance of accurately capturing architectural information down to the lower systems levels, where a majority of the system architecture information resides.

4.4.4. Suitability of Artifacts for Information Capture

Based on the results presented in Table 7, Table 8, Table 9, and Table 10, the SysML MBSE architecture artifact types generated using the MBSE approach applied to COM system in this research (Glossaries, Profiles, Block Definition Diagrams, Internal Block Diagrams, Activity Diagrams, Requirements Diagrams) were more suited for capturing the architecture information relative to the non-MBSE artifact types (Spreadsheets, Presentations, and Manuscripts). Table 7 and Table 9 show that both the SysML MBSE and non-MBSE artifact types all captured some architecture information with good alignment. However, Table 10 shows that all three non-MBSE artifact types (100% of the artifact types) had inadequacies in their ability to cover architecture information, whereas Table 8 shows that only two out of the 6 types of SysML MBSE artifacts (Block Definition Diagrams and Requirements Diagrams) (33% of the MBSE

artifact types) had inadequacies in their ability to cover architecture information. The better suitability of the MBSE artifact types for capturing architecture information is partly due to the MBSE tool being specifically developed with the specialized constructs needed to capture systems engineering-relevant knowledge.

TABLE 7. NUMBER OF COM SYSTEM ARCHITECTURE SYSTEM KNOWLEDGE TYPE ALIGNMENTS PER MBSE ARTIFACT ORGANIZED BY KNOWLEDGE CATEGORY. ALIGNMENTS INDICATE THAT THE ARTIFACT PROPERLY CAPTURES KNOWLEDGE TYPES FOR THE KNOWLEDGE CATEGORY. GREEN INDICATES A HIGH NUMBER OF KNOWLEDGE TYPE ALIGNMENTS, YELLOW INDICATES A MEDIUM NUMBER OF KNOWLEDGE TYPE ALIGNMENTS, AND RED INDICATES A LOW NUMBER OF KNOWLEDGE TYPE ALIGNMENTS RELATIVE TO THE HIGHEST NUMBER OF KNOWLEDGE SUBCATEGORY ALIGNMENTS FOR EACH MBSE ARTIFACT.

	Category		Subcategory		Glossary	Profile	BDD	IBD	Activity	Requirement
Knowledge	A	Factual	A.A	Terminology	1	0	0	0	0	0
			A.B	Specific Details/Elements	1	0	8	8	2	9
	B	Conceptual	B.A	Classifications/Categories	0	4	2	1	0	0
			B.B	Principles/Generalizations	0	0	1	0	0	0
			B.C	Theories/Models/Structures	0	0	4	1	2	1
	C	Procedural	C.A	Subject-specific Skills/Algorithms	0	0	0	0	2	0
			C.B	Subject-specific Techniques/Methods	0	0	0	0	0	0
			C.C	Criteria for Procedure Use	0	0	0	0	0	0
	D	Metacognitive	D.A	Strategies	0	0	0	0	0	0
			D.B	Cognitive Tasks	0	0	0	0	0	0
			D.C	Self-knowledge	0	0	0	0	0	0
				Total	2	4	15	10	6	10

TABLE 8. NUMBER OF COM SYSTEM ARCHITECTURE SYSTEM KNOWLEDGE TYPE NON-ALIGNMENTS PER MBSE ARTIFACT ORGANIZED BY KNOWLEDGE CATEGORY. NON-ALIGNMENTS INDICATE THAT THE ARTIFACT DOES NOT PROPERLY CAPTURE KNOWLEDGE TYPES FOR THE KNOWLEDGE CATEGORY. RED INDICATES A HIGH NUMBER OF KNOWLEDGE TYPE NON-ALIGNMENTS, YELLOW INDICATES A MEDIUM NUMBER OF KNOWLEDGE TYPE NON-ALIGNMENTS, AND GREEN INDICATES A LOW NUMBER OF KNOWLEDGE TYPE NON-ALIGNMENTS RELATIVE TO THE HIGHEST NUMBER OF KNOWLEDGE SUBCATEGORY NON-ALIGNMENTS FOR EACH MBSE ARTIFACT

	Category		Subcategory		Glossary	Profile	BDD	IBD	Activity	Requirement
Knowledge	A	Factual	A.A	Terminology	0	0	0	0	0	0
			A.B	Specific Details/Elements	0	0	0	0	0	0
	B	Conceptual	B.A	Classifications/Categories	0	0	0	0	0	0
			B.B	Principles/Generalizations	0	0	1	0	0	0
			B.C	Theories/Models/Structures	0	0	0	0	0	0
	C	Procedural	C.A	Subject-specific Skills/Algorithms	0	0	0	0	0	0
			C.B	Subject-specific Techniques/Methods	0	0	1	0	0	0
			C.C	Criteria for Procedure Use	0	0	1	0	0	0
	D	Metacognitive	D.A	Strategies	0	0	1	0	0	1
			D.B	Cognitive Tasks	0	0	2	0	0	0
			D.C	Self-knowledge	0	0	1	0	0	0
				Total	0	0	7	0	0	1

TABLE 9. NUMBER OF COM SYSTEM ARCHITECTURE SYSTEM KNOWLEDGE TYPE ALIGNMENTS PER NON-MBSE ARTIFACT ORGANIZED BY KNOWLEDGE CATEGORY. ALIGNMENTS INDICATE THAT THE ARTIFACT PROPERLY CAPTURES KNOWLEDGE TYPES FOR THE KNOWLEDGE CATEGORY. GREEN INDICATES A HIGH NUMBER OF KNOWLEDGE TYPE ALIGNMENTS, YELLOW INDICATES A MEDIUM NUMBER OF KNOWLEDGE TYPE ALIGNMENTS, AND RED INDICATES A LOW NUMBER OF KNOWLEDGE TYPE ALIGNMENTS RELATIVE TO THE HIGHEST NUMBER OF KNOWLEDGE SUBCATEGORY ALIGNMENTS FOR EACH NON-MBSE ARTIFACT

	Category		Subcategory		Spreadsheet	Presentation	Manuscript
Knowledge	A	Factual	A.A	Terminology	1	0	0
			A.B	Specific Details/Elements	12	10	6
	B	Conceptual	B.A	Classifications/Categories	0	1	0
			B.B	Principles/Generalizations	0	0	0
			B.C	Theories/Models/Structures	0	3	0
	C	Procedural	C.A	Subject-specific Skills/Algorithms	1	0	0
			C.B	Subject-specific Techniques/Methods	0	0	0
			C.C	Criteria for Procedure Use	0	0	0
	D	Metacognitive	D.A	Strategies	0	0	0
			D.B	Cognitive Tasks	0	0	0
			D.C	Self-knowledge	0	0	0
				Total	14	14	6

TABLE 10. NUMBER OF COM SYSTEM ARCHITECTURE SYSTEM KNOWLEDGE TYPE NON-ALIGNMENTS PER NON-MBSE ARTIFACT ORGANIZED BY KNOWLEDGE CATEGORY. NON-ALIGNMENTS INDICATE THAT THE ARTIFACT DOES NOT PROPERLY CAPTURE KNOWLEDGE TYPES FOR THE KNOWLEDGE CATEGORY. RED INDICATES A HIGH NUMBER OF KNOWLEDGE TYPE NON-ALIGNMENTS, YELLOW INDICATES A MEDIUM NUMBER OF KNOWLEDGE TYPE NON-ALIGNMENTS, AND GREEN INDICATES A LOW NUMBER OF KNOWLEDGE TYPE NON-ALIGNMENTS RELATIVE TO THE HIGHEST NUMBER OF KNOWLEDGE SUBCATEGORY NON-ALIGNMENTS FOR EACH NON-MBSE ARTIFACT.

	Category		Subcategory		Spreadsheet	Presentation	Manuscript
Knowledge	A	Factual	A.A	Terminology	0	0	0
			A.B	Specific Details/Elements	0	0	0
	B	Conceptual	B.A	Classifications/Categories	1	5	0
			B.B	Principles/Generalizations	0	1	1
			B.C	Theories/Models/Structures	3	1	1
	C	Procedural	C.A	Subject-specific Skills/Algorithms	1	0	0
			C.B	Subject-specific Techniques/Methods	0	0	1
			C.C	Criteria for Procedure Use	0	0	1
	D	Metacognitive	D.A	Strategies	1	0	1
			D.B	Cognitive Tasks	0	0	2
			D.C	Self-knowledge	0	0	1
				Total	6	7	8

Table 7 also highlights where some of the deficiencies lie within the specific MBSE approach using Cameo Systems Modeler and SysML, specifically within the block definition diagrams (BDDs). The misalignments shown represent poor coverage of design principles, design approaches and rationales, risk, development strategies and rationales, organizational core competencies, and requirement verification methods.

4.4.5. Future Work

Future work should include testing the methodology on additional systems, testing the methodology with different MBSE languages, tools, and approaches, as well as developing methods and constructs to more accurately represent high-level knowledge:

- Testing the methodology on additional systems: Applying the MBSE and non-MBSE approaches on additional systems can help validate the methodology across additional engineering domains, as well as gather additional data on the types of system knowledge captured and how well the MBSE approach is able to capture the knowledge. Data from more complex systems that span multiple engineering domains, utilize a higher number of design patterns, and execute a larger series of behaviors would further demonstrate the advantages of MBSE approaches over the non-MBSE approaches
- Testing the methodology with different MBSE languages and tools: Different languages possess unique ontologies with knowledge constructs that have to potential to capture more higher-levels of knowledge. Additionally, modeling tools vary in their implementation of the MBSE languages, ability to generate extensions, and possession of unique, tool-specific features used to capture and organize system knowledge.
- Developing methods and constructs to more accurately represent high-level knowledge categories: The current MBSE approach using SysML and Cameo Systems Model did not fully capture high-level knowledge, such as design principles, design approaches and rationales, risk, development strategies and rationales, organizational core competencies, and requirement verification methods. Future work should focus on developing methods and constructs, as well as exploring additional languages, to more accurately represent these types of knowledge in order to improve the MBSE approach. New methods, language constructs, and tools could improve the approach's ability to properly capture this high-level system knowledge not currently capture in the SysML-based MBSE approach implemented in Cameo Systems Modeler

4.5. Conclusion

A model-based systems engineering (MBSE) approach implemented with Cameo Systems Modeler using SysML was applied to architecting an orbiting sample Capture and Orient Module (COM) system concept for a Capture, Containment, and Return System (CCRS) payload concept for potential Mars Sample Return (MSR). An architecture framework was established, covering three organization levels of the system, along with structural, behavioral, data, and requirements perspectives. The COM system architecture was captured in parallel using both MBSE and non-MBSE approaches. A total of 4,389 knowledge elements were classified using the Revised Bloom's Taxonomy knowledge dimension and used to quantitatively compare the two approaches. Overall, the MBSE approach more completely captured architectural knowledge than the non-MBSE, document-based approach. A majority of these benefits were seen in the high-level conceptual and procedural knowledge categories, where the MBSE approach provided a means to more accurately represent categorical information through the definition of blocks and stereotypes, abstract structural relationships through the definition of directed compositions, component models through functional allocations, and procedural knowledge through the definition of control flows.

Limitations to the MBSE approach implemented with Cameo Systems Modeler using SysML resided within its ability to fully represent certain conceptual, procedural, and metacognitive categories such as design principles, design approaches and rationales, risks, development strategies and rationales, organizational core competencies, and requirement verification methods. Future work should focus on developing methods and constructs, as well as exploring additional languages, to more accurately represent these types of knowledge in order to improve

the MBSE approach, with a focus on improving the depth of knowledge that can be encoded in structural representations.

Overall, relative to document-based systems engineering approaches, the MBSE approach more completely captured architecture knowledge, which can potentially help reduce risk of information loss or mistranslation during information encoding and retrieval that can occur with standard documentation approaches. These results help to demonstrate the benefits of MBSE in managing a complex robotic space system and strengthen the case for adopting MBSE within the broader systems engineering community.

The COM system architecture framework and architecture information content quantities developed to assess the improvement in information capture of the COM architecture descriptive model using MBSE, which is the goal of Research Question 1, can now be used to assess the implementation effort improvements of the COM architecting process using MBSE, which is the goal of Research Question 2. Chapter 5 uses the COM system architecture framework to develop a system architecting process to architect a robotic space system, which consists of a series of 76 tasks to systematical define the COM architecture across each of the four perspectives and down each of the three organization levels. The architecture information content is used to quantify the amount of information transferred between these process tasks for both an MBSE and non-MBSE architecting approach. The benefits provided by the MBSE architecting process approach over the non-MBSE architecting process approach are then assessed based on how much of this information can be transferred automatically vs. manually through leveraging MBSE resources.

5. COMPARATIVE ANALYSIS OF MODEL-BASED AND TRADITIONAL SYSTEMS ENGINEERING APPROACHES FOR ARCHITECTING A ROBOTIC SPACE SYSTEM THROUGH AUTOMATIC INFORMATION TRANSFER[2]

5.1. Introduction

On August 5, 2012, NASA's 900 kg Mars Science Laboratory (MSL) Curiosity rover successfully landed on the surface of Mars and set out to search for evidence of past habitable environments [116] [2]. The Curiosity rover pushed the boundaries of technology and systems engineering, consisting of approximately 50,000 parts, involving nearly 3,000 NASA employees and 4,000 non-government workers, and was considered the most complex rover of its time ever sent to another planet [116] [3] [4].

Despite the technical and scientific achievements of the rover, the project experienced numerous development challenges, and in the end, saw an increase in over $881 million in costs from its original 2008 project baseline, as well as a 26-month launch delay due to technical problems that necessitated late design changes in hardware, avionics, and software [5]. A metric for design changes used by NASA is "drawing growth" after the Critical Design Review (CDR), where MSL saw a 147% growth [6]. Some of these late design changes were attributed to the discovery of divergent requirements uncovered late during the testing phase. These divergent requirements were found to be a consequence of not having a rigorously defined architecture to pull together and cohesively manage the complex web of documentation of system and

[2] This chapter was submitted to *IEEE Access*. The citation is as follows: Younse, P., J. Cameron, and T.H. Bradley, "Comparative Analysis of Model-based and Traditional Systems Engineering Approaches for Architecting a Robotic Space System through Automatic Information Transfer," submitted to *IEEE Access*, 2021.

subsystem functional requirements, environmental requirements, interface control documents, institutional policy documents, and planetary protection requirements [7].

The complexity of space missions is quickly growing faster than NASA's ability to manage them [21]. Two future space missions under development by NASA and the European Space Agency (ESA) are the Sample Retrieval Lander (SRL) and Earth Return Orbiter (ERO) missions, which are planned as follow-up missions to the Mars 2020 rover mission as part of the MSR campaign (see Figure 19) [22] [117] [18] [19] [20]. SRL would land on Mars with a fetch rover to retrieve samples collected by the Mars 2020 rover and place them into Mars orbit within an Orbiting Sample (OS) container. The ERO would autonomously capture the OS and return it to Earth within an Earth Entry Vehicle (EEV). An independent review board reviewed the Pre-Phase A MSR technical concept and found the architecture extremely complex, requiring a long series of critical events to be carried out with high precision and reliability [118]. If NASA is to succeed in future complex robotic space missions like those associated with MSR, new systems engineering approaches to manage the growth in complexity associated with these future missions could play a critical role in controlling costs, maintaining schedule, and ensuring mission success.

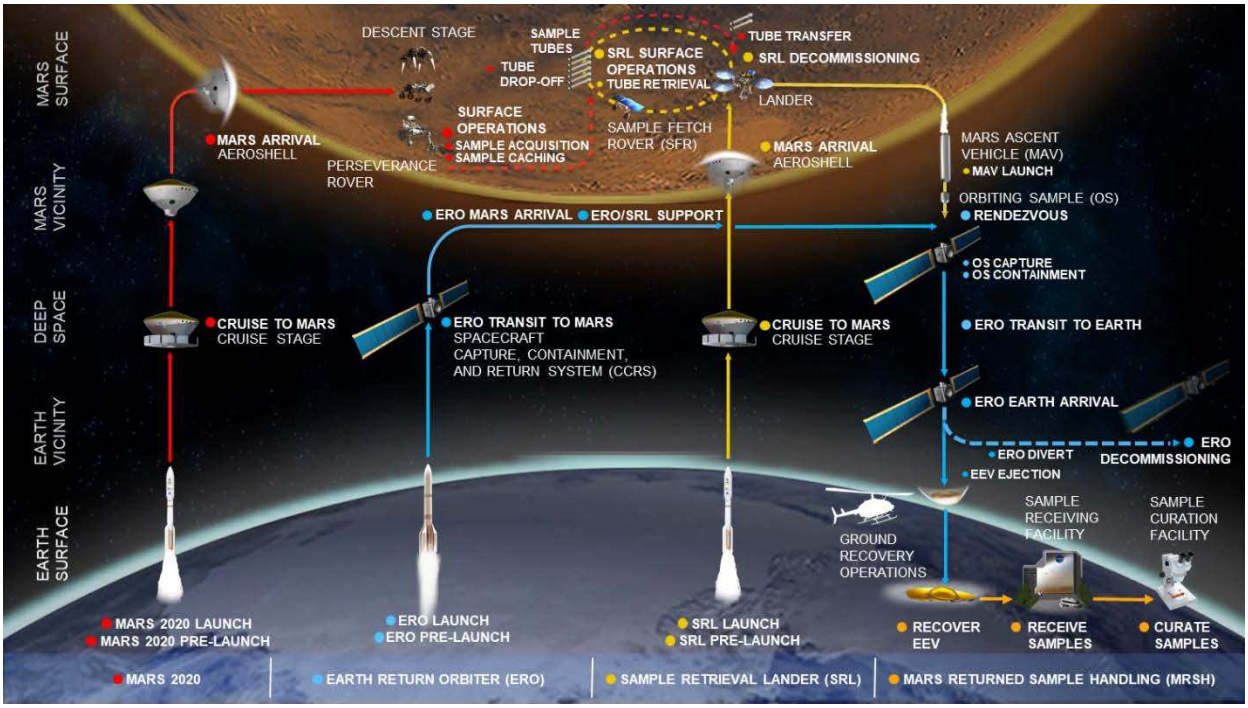


FIGURE 19. NOTIONAL MSR ARCHITECTURE. NOTE THAT ALL ELEMENTS BEYOND MARS 2020 ARE CONCEPTUAL [22].

MBSE provides a systems engineering paradigm to manage complex systems by aiming to reduce design errors, reduce cost through prevention of costly rework, and improve system quality and project performance over traditional systems engineering techniques [23] [24]. MBSE helps to achieve this during the architecting process by developing an integrated system model that captures key system architectural information and links that information through model associations.

The purpose of this research is to explore the advantages of an MBSE approach in architecting a robotic space system relative to a non-MBSE approach, as assessed by the quantity of information transfer that can be automated for carrying out the architecting process. One of the major drivers of project cost, schedule overruns, and project risk is process iteration [119]. Several causes of process iterations include poor communication of information (e.g., information not clearly or appropriately transmitted) and errors (e.g., defective information

created and propagated without correction). Automation of information transfer has the potential to improve communication of information and reduce errors that could later lead to costly process iterations, rework, and design changes.

The MSR CCRS COM Pre-Phase A architectural development activity was used as a case study to assess the benefits of MBSE over traditional systems engineering in achieving automatic information transfer.

This research is motivated by the observation that despite the claims made in the literature that MBSE is beneficial to the development of engineered systems, there is lack of empirical evidence in the literature that supports this hypothesis [120]. Additionally, there is a limited number of case studies with side-by-side comparison of MBSE and non-MBSE approaches that provide quantitative evidence of the advantages MBSE approaches over traditional, document-based approaches [24]. This paper presents a unique methodology that uses DSMs as a tool to perform a side-by-side comparison of an MBSE and non-MBSE architecting approach used to architect the COM and quantitatively measures the benefits in terms of automatic information transfer between activities within the DSMs. This research provides the following unique contributions to the literature:

- A case study that compares an MBSE approach side-by-side with a non-MBSE approach for architecting a robotic space system
- An MBSE approach for architecting a robotic space system that defines the structure, data, behavior, and requirements of the system at each organization level and incorporates trade studies and peer reviews between levels
- A methodology that uses DSMs as a tool to quantitatively measure the benefits of an MBSE approach relative to a non-MBSE approach

- Quantitative evidence of the benefits of an MBSE approach over a non-MBSE approach for architecting a robotic space system in terms of automatic information transfer

This paper is organized as follows. Section II provides background on MBSE, architecture frameworks, design structure matrices, and the COM case study. Section III provides details on the methodology, consisting of developing a resource breakdown structure, synthesizing a system architecting process, mapping out architecting process tasks of the non-MBSE and MBSE approaches, recording the quantities of information transfer between tasks, comparing quantities of automatic information transfer between the two approaches, and extending the MBSE architecting approach to increase automatic information transfer. Section IV provides a summary of the results. Section V discusses the benefits of MBSE based on an analysis of the results, as well as recommendations for future work. Section VI finishes with a conclusion.

5.2. Background

The following section provides an overview of MBSE, architecture frameworks, DSMs and the COM robotic space system used as a case study in this research.

5.2.1. Model-based Systems Engineering

MBSE emphasizes the use of models to perform systems engineering activities that are traditionally performed using documents [25]. To carry out MBSE, a modeling language, modeling method, and modeling tool are needed [35]. Listings of MBSE modeling languages, methods, and tools can be found in publications by Rashid et al. [121], Estefan [122], Madni et al. [123], and the Body of Knowledge and Curriculum to Advance Systems Engineering (BKCASE) editorial board [124]. MBSE aims to provide benefits over traditional, document-based systems engineering in terms of reduced effort to implement system development through

increased productivity, reduced inefficiencies, and reduced lag in information flow [24] [51] [54] [46].

MBSE methods have been used to aid in the development of complex space systems. Examples of space system projects that utilized MBSE include the ExoMars mission [47] and e.Deorbit mission [61].

Mazzini et al. assessed the applicability of the Model-Based Space System Engineering (MBSSE) methodology on the ExoMars mission [47]. System requirements, system context, data, control flows, mission use cases, scenarios, functional architecture, and software architecture were modeled. The MBSSE approach proved successful in defining a preliminary space system and offered improved traceability and separation of concerns between systems engineering and software engineering.

Estable et al. applied a Federated and Executable Models MBSE methodology during the architecture definition phase of the ESA e.Deorbit robotic satellite mission study [61]. SysML models were developed to capture the mission Concept of Operations, system capabilities, functional architecture, safety diagram, fault tree, product tree, and requirements using Cameo Systems Modeler. The methodology demonstrated improved efficiency in the systems engineering work.

The above case studies asserted that there are benefits to using MBSE when applied to space systems. However, none of these studies provided quantitative evidence of MBSE's advantages over traditional, non-MBSE approaches. Additionally, there is an absence of case studies that perform side-by-side comparisons of an MBSE approach with a non-MBSE (traditional, document-based) approach [24]. This research aims to address these gaps in the literature by providing a case study that performs side-by-side comparisons of MBSE and non-MBSE

approaches, as well as collect quantitative data that can be used to measure the costs and benefits of the two approaches. This paper specifically investigates how an MBSE approach can improve the systems engineering process by automating the transfer of information between tasks.

5.2.2. Architecture Framework

Architecture can be defined as the fundamental concepts or properties of a system in its environment embodied in its elements, relationships between those elements, and principles of their design and evolution [125]. An architecture framework collects and relates viewpoints to enable the system architect to construct useful and consistent architecture descriptions [95].

The framework used in this research to collect and relate the architectural information content for the COM is shown in Figure 20. The framework was synthesized from frameworks used in the Model-Based System Architecture Process (MBSAP) [30] and MagicGrid [44] methodologies. A table format, like that used by MagicGrid, was adopted due to its ability to visually represent and organize architectural artifacts. The structure, data, behavior, and requirements perspectives from the MBSAP perspectives were applied to the columns of the table. The system levels were applied to the rows of the table. Three levels were defined for the COM framework: Module Level (Level 4), Subsystem Level (Level 5), and Assembly Level (Level 6). These levels were defined based on the CCRS project's product breakdown structure.

The architectural information content (e.g., system elements, element properties, and relationships to describe the architecture) captured within the COM architecture framework is also summarized in Figure 20. This information content was selected by the COM engineering team at JPL and informed by the NASA Systems Engineering Handbook [77], Expanded Guidance for NASA Systems Engineering [111], and MBSAP [30].

		Perspective			
		Structure	Data	Behavior	Requirements
Organization Level	Module Level (L4)	• **Glossary:** Terms, Term Descriptions • **Classifications:** Hierarchy Levels, Interface Types, Element Ownerships • **Product Breakdown:** Product Breakdown Structure, Element Types, Element Hierarchy Levels, Element Ownership Assignments, Structural Decompositions • **Block Diagrams:** System Block Diagram, Parts, Part Types, Part Ownerships, Part Ownership Assignments, Interfaces, Interface Type Assignments, Flow Item Allocations, Flow Directions • **Specifications:** Description, Owner, Attributes, Allocated Functions, Data Generation and Use, External Interfaces, Allocated Requirements, Applicable Design Principles, Recommended Approaches, Approach Rationales, Risks, Development Strategies, Development Strategy Rationales, Core Competencies	• **Glossary:** Terms, Term Descriptions • **Classifications:** Hierarchy Levels • **Data Model:** Hierarchical Data Model, Data Elements, Data Element Hierarchy Levels, Specializations	• **Glossary:** Terms, Term Descriptions • **Scenarios:** Functional Flows, Functional Decompositions, Functions, Function Hierarchy Levels, Control Flows, Functional Allocations	• **Glossary:** Terms, Term Descriptions • **Classifications:** Hierarchy Levels, Requirement Types • **Requirements:** Requirements Hierarchy, Requirements, Requirement IDs, Requirement Hierarchy Levels, Requirement Requirement Types, Requirement Texts, Requirement Rationales, Requirement Types, Verification Methods, Parents, Requirements Allocations
	Subsystem Level (L5)				
	Assembly Level (L6)				

FIGURE 20. ARCHITECTURE FRAMEWORK USED FOR DEVELOPING THE COM ARCHITECTURE WITH A LISTING OF THE SYSTEM INFORMATION CONTENT FOR ALL LEVELS WITHIN EACH PERSPECTIVE.

5.2.3. Design Structure Matrix

A DSM is a network modeling tool used to graphically represent elements of a system and their interactions in the form of a matrix [119]. Process Architecture DSMs model process activities as the elements, and flows of information between the activities as the interactions. The "inputs in rows" (IR) convention was used, where inputs to activities are captured in the rows, and outputs from activities are captured in the columns. The numerical DSM extended form was used, where numerical values and colors represent the number and type of interactions. Cordero et al. used numeric DSMs to sequence, analyze, and improve model-based concurrent conceptual design processes in conceptual design studies of space missions [126]. Similarly, DSMs were used in this research to model and analyze the MBSE and non-MBSE architecting approaches used to develop the COM architecture.

5.2.4. Capture and Orient Module Case Study

The COM Pre-Phase A architectural development was used as a case study to assess the benefits of the MBSE approach over the traditional, non-MBSE systems engineering approach.

The CCRS houses the COM. The COM captures, constrains, orients, inspects, and assembles the OS into the Primary Containment Vessel (PCV) in preparation for PCV sealing and installation into the EEV for future delivery to Earth.

An early concept for CCRS with the COM is shown in Figure 21. The COM is organized into three architectural levels of decomposition: Level 4 (Module Level), Level 5 (Subsystem Level), and Level 6 (Assembly Level). Level 4 represents the overall COM. Level 5 represents the seven major COM subsystems: Capture Mechanism (CM), Sensor System (SS), Capture Cone (CC), Orientation Mechanism (OM), Transfer Mechanism (TM), COM Infrastructure (CI), and Thermal Control System (TCS). These subsystems are illustrated in Figure 22. Level 6 represents the assemblies that make up each of the subsystems, such as individual actuators, mechanisms, structural elements, and sensors.

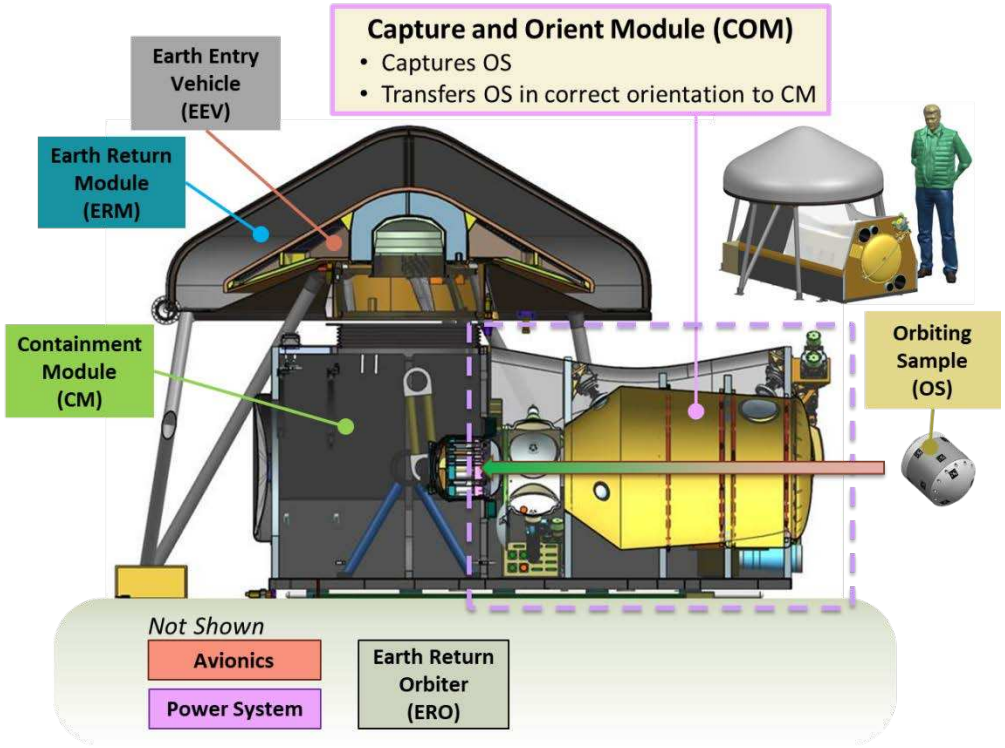


FIGURE 21. NOTIONAL CCRS CONCEPT [22].

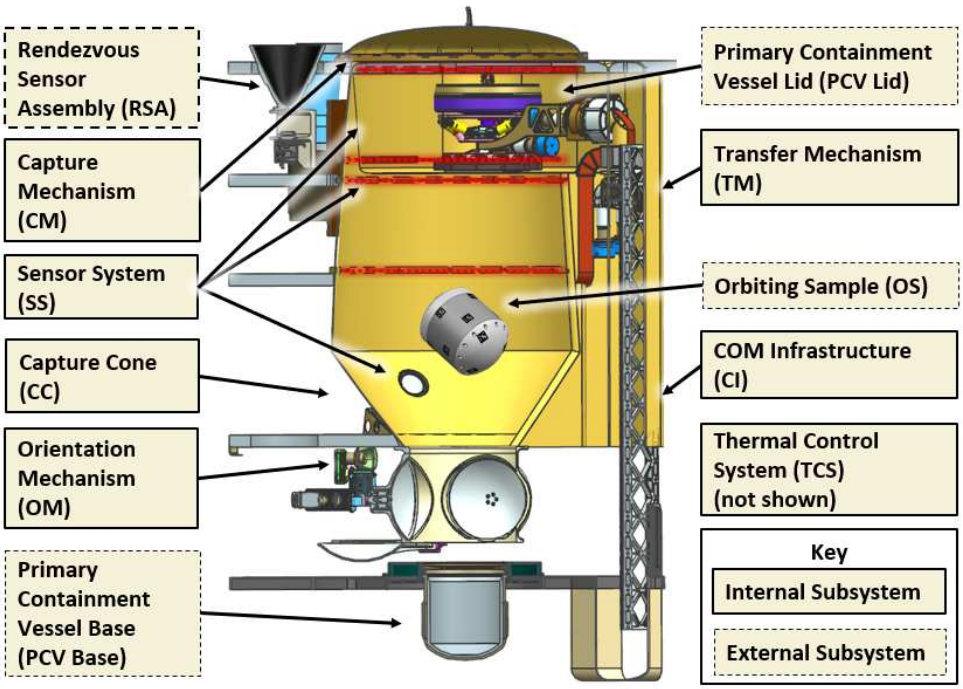


FIGURE 22. NOTIONAL COM CONCEPT [22].

The COM would be designed to operate autonomously in Mars orbit within a space environment. Figure 23 shows an operational concept of the COM, depicting OS capture, constraint, orientation, inspection, and assembly into the PCV. Further details on the COM and its operations are described by Younse et al. [22].

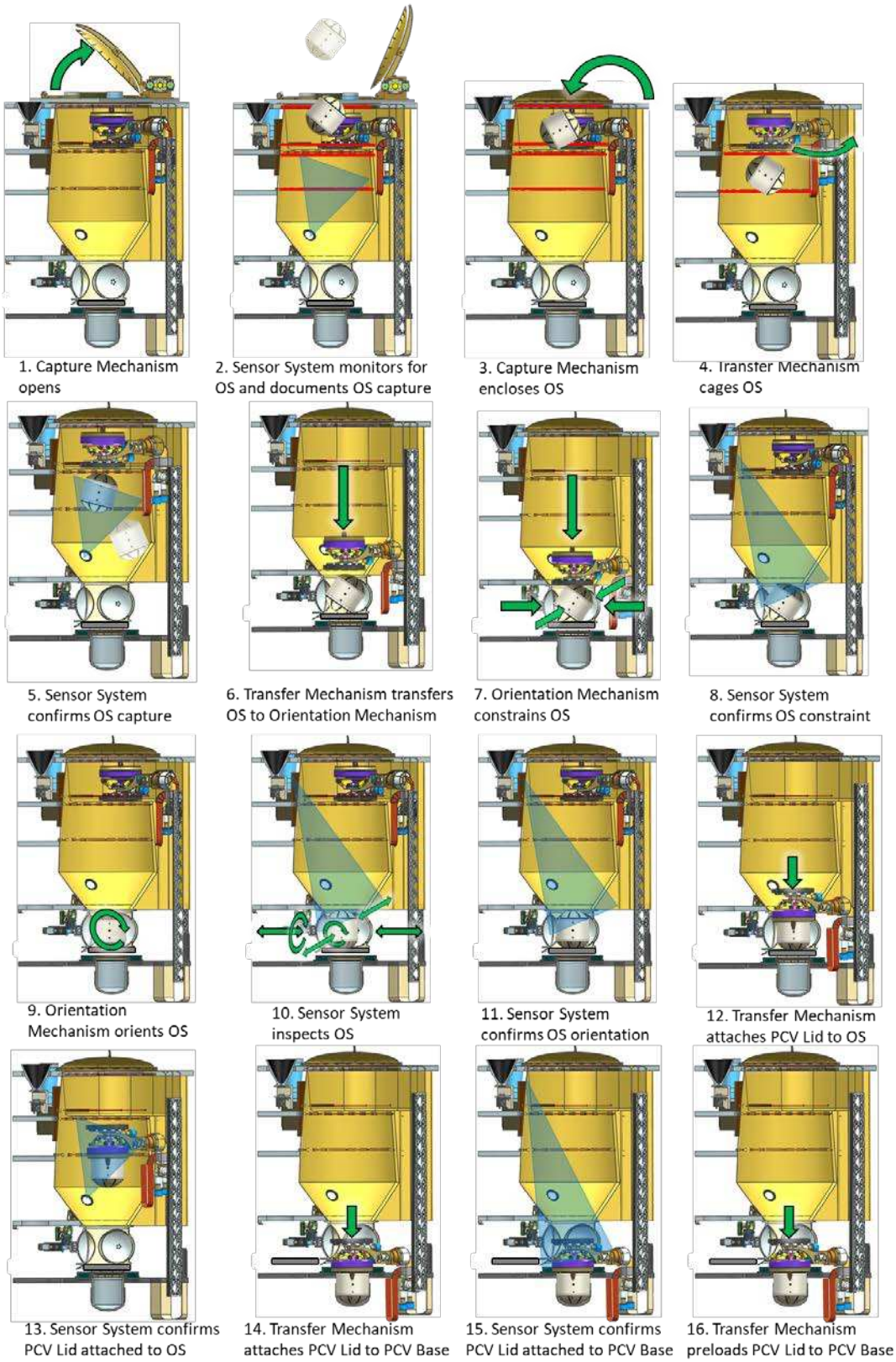


FIGURE 23. COM OPERATIONAL CONCEPT [22].

The COM architecture development took place during Pre-Phase A, which is the first of seven phases of the NASA Project Life-Cycle. During Pre-Phase A, the project explores a range of ideas and develops an initial, feasible system concept [78]. The architecture development was carried out over the course of two years by the COM engineering team at the Jet Propulsion Lab (JPL) in Pasadena, California. The same team members carried out the architecting tasks using both the MBSE and non-MBSE approaches in parallel over the same course of time.

5.3. Methodology

The DSM Approach to Architectural Modeling and Analysis described by Eppinger and Browning [119] was followed to model and analyze the architecting process. The approach involves decomposing the system process down to its constituent elements, identifying the relationships amongst the system's elements, analyzing the elements and their relationships and their implications for the system process, displaying the system process in the form of a DSM model, and improving the system process based on the results of the DSM analysis. This DSM approach was further elaborated and specialized for architecting a robotic space system, and carried out through a series of seven steps described in this section.

5.3.1. Step 1: Develop a Resource Breakdown Structure for Architecting a Robotic Space System

Resource breakdown structures (RBS) were developed for both the MBSE and non-MBSE architecture approaches. Resources were defined as any person or systems engineering artifact generated and utilized to capture and organize architecture information, as well as carry out tasks associated with the architecting process. For the COM, this included specific JPL personnel and specialized systems engineering artifacts capable of capturing and communicating robotics,

mechanical, electrical, thermal, flight software, contamination control, and planetary protection technical knowledge associated with a robotic space system.

Microsoft PowerPoint, Word, and Excel were utilized to implement the non-MBSE tasks due to their compatibility with current document-based systems engineering artifacts used at JPL, as well as common use within the engineering team for capturing and communicating numerical, textual, and graphical system information.

Cameo Systems Modeler using the SysML language profile was utilized for the MBSE approach to allow model integration into the top-level MSR campaign model that was previously established by JPL and ESA to integrate technical and programmatic information across all missions and mission elements [113]. SysML and MagicDraw (the prior release of Cameo Systems Modeler) was chosen by the campaign as the implementing modelling language and tool due to the extensive experience and resources of both agencies [114].

5.3.2. Step 2: Synthesize a Robotic Space System Architecting Process

A system architecting process was developed to architect a robotic space system and generate the artifacts developed in Step 1 using the COM architecture framework shown in Figure 20. The general architecting process flow was developed following the MBSAP methodology, where the top level of the system is defined in an Operational Viewpoint, the lower levels in the Logical/Functional Viewpoint, and final level of the design in the Physical Viewpoint [30]. A layered approach to define the architecture following the STRATA methodology [41] and NASA Systems Engineering Engine System Design Process [77] was taken, where each level of the architecture is defined prior to drilling down to the next lower, more specific levels.

An object-oriented design methodology was taken to define each architectural level. In object-oriented design, the first step is to identify objects comprising the system, their

associations and characteristics, and their interactions and interfaces [30]. This information is captured in the Structural and Data perspectives. Next, sequences of activities carried out by the individual objects are defined, which are captured in the Behavioral perspective. Requirements are developed as the final step of each level, as performed in the SCARIT Process Model [127].

5.3.3. Step 3: Map out the Robotic Space System Architecting Process Tasks of a Non-MBSE Approach

For the non-MBSE architecting approach, the tasks defined in Step 2 were sequenced in a series of activity diagrams, with the resources represented as swimlanes. Tasks were allocated to personnel resources. Information generated and consumed by each task were identified and traced to each document resource (i.e. slides, manuscripts, spreadsheets, prototypes). Figure 24 provides an example of an activity diagram showing two tasks allocated to the "COM Systems Engineer." Information generated from the tasks are depicted in the boxes leading down into the document where the information is captured. Information consumed by the tasks are depicted in the boxes leading from the resource where the information is retrieved. Information transferred from one task to another task occurs when the same information is recorded to and retrieved from the same resource. In this example, the information type "Element Types" are transferred from the Define CCRS Structural Elements task to the Define CCRS Internal Structure task. The quantities of information elements transferred from one task to another in this manner are what were captured and quantified in the DSMs.

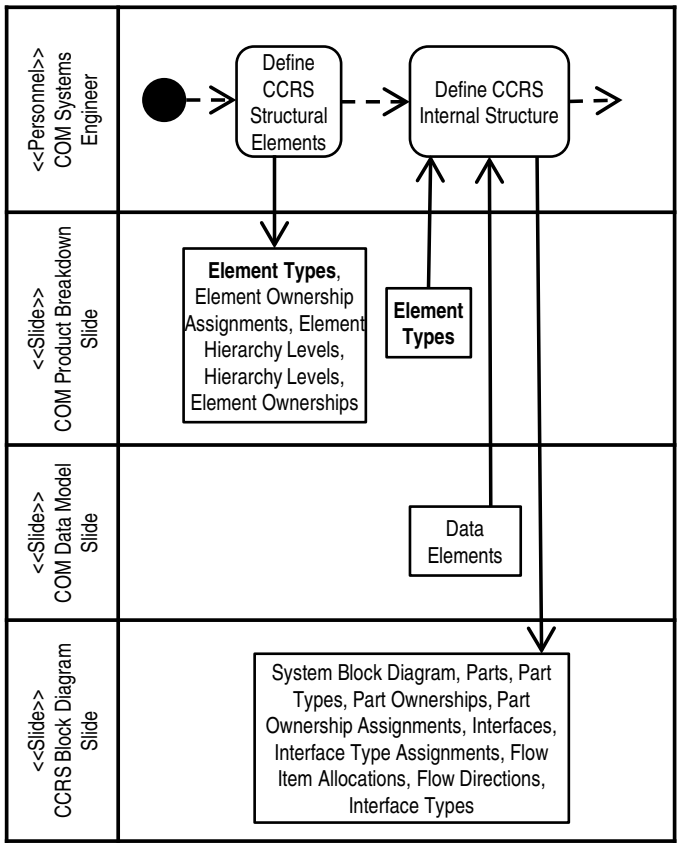


FIGURE 24. EXAMPLE ACTIVITY DIAGRAM FOR THE NON-MBSE APPROACH FOR TASK 1 AND TASK 2 FROM TABLE 11 SHOWING RESOURCES, INTERACTIONS BETWEEN RESOURCES, AND INFORMATION GENERATION AND CONSUMPTION FOR EACH TASK. THE INFORMATION TYPE "ELEMENT TYPES" IS BOLDED TO SHOW AN EXAMPLE OF INFORMATION TRANSFER BETWEEN THE TWO TASKS.

5.3.4. Step 4: Map out the Robotic Space System Architecting Process Tasks of an MBSE Approach

The tasks defined in Step 2 were also sequenced in a series of activity diagrams for the MBSE approach. Information elements generated and consumed by each task were identified and traced to each document or model resource (i.e. slides, spreadsheets, prototypes, block definition diagrams, internal block diagrams, blocks, activity diagrams, requirements diagrams, profiles, glossary tables). Figure 25 shows an example of an activity diagram for the first two architecting process tasks for the MBSE approach.

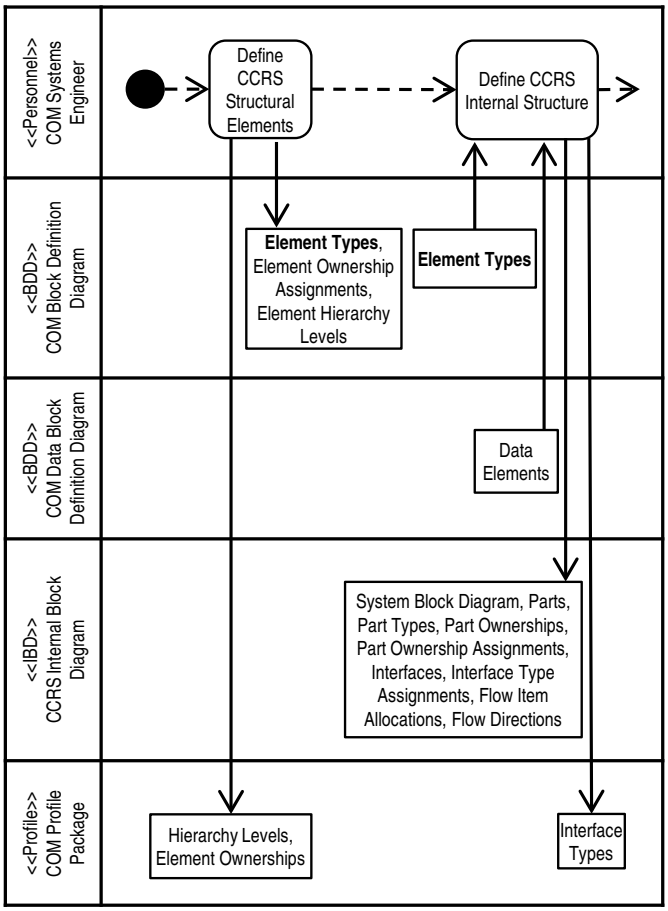


FIGURE 25. EXAMPLE ACTIVITY DIAGRAM FOR THE MBSE APPROACH FOR TASK 1 AND TASK 2 FROM TABLE 11 SHOWING RESOURCES, INTERACTIONS BETWEEN RESOURCES, AND INFORMATION GENERATION AND CONSUMPTION FOR EACH TASK. THE INFORMATION TYPE "ELEMENT TYPES" IS BOLDED TO SHOW AN EXAMPLE OF INFORMATION TRANSFER BETWEEN THE TWO TASKS.

5.3.5. Step 5: Record the Quantities of Information Transfer between the Robotic Space System Architecting Tasks of the Non-MBSE and MBSE Approaches

DSMs were developed for both the non-MBSE and MBSE approaches to describe the information transferred between tasks (as mapped out in Steps 3 and 4), classify the information transfer based on whether it is manual or automatic, and quantify the amount of information manually or automatically transferred.

Figure 26 shows an example DSM for 16 of the 76 COM architecting process tasks that cover the COM Architecture Definition, COM Trade Study, and COM Architecture Peer Review. The task IDs are represented on both the vertical and horizontal axes of the matrix. Information fed forward between tasks is captured in the lower left portion of the matrix. Information fed back between tasks is captured in the upper right portion of the matrix. Information transferred manually between resources (information that must be manually transcribed or recreated between resources due to the lack of explicit links and associations of information elements between model views or documents) is indicated by the red-shaded cells. Information transferred automatically (information that is automatically filled in or updated between resources due to explicit links and associations of information elements between model views) is indicated by the green-shaded cells. The quantities of information transferred between tasks are represented as numbers within the cells. Cells that are shaded and show a zero can have information transferred, but did not have any in those instances.

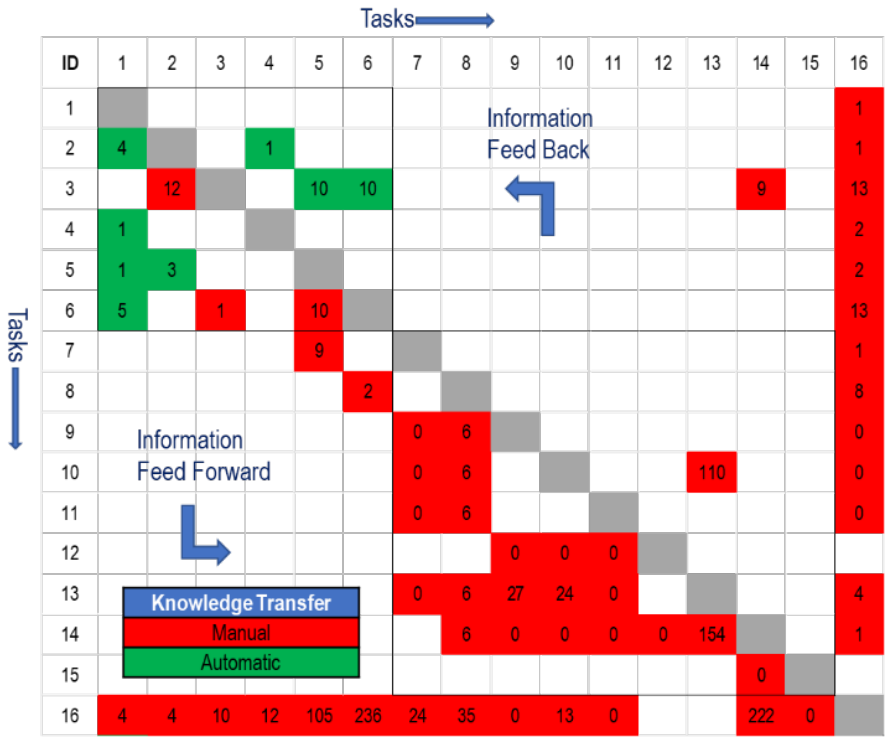


FIGURE 26. EXAMPLE DESIGN STRUCTURE MATRIX FOR THE MBSE APPROACH FOR THE L4 ARCHITECTURE DEFINITION, COM TRADE STUDY, AND COM ARCHITECTURE PEER REVIEW TASKS.

5.3.6. Step 6: Compare the Quantities of Automatic Information Transfers between the Non-MBSE and MBSE Robotic Space System Architecting Approaches

The number of manual and automatic information transfer between tasks were compared in a table for both the non-MBSE and MBSE approaches. These values were taken from each of the DSMs, and presented both in terms of individual quantities and percentages of the total quantities of information transfer.

5.3.7. Step 7: Extend the MBSE Robotic Space System Architecting Approach

Based on the results of the DSM analysis performed in Step 6, an extensive MBSE approach was defined that further takes advantage of MBSE resources for the trade study and peer review tasks to further increase automatic information transfer. This involved looking at what tasks were

manual, determining which MBSE resources could be further utilized for these tasks, developing a DSM of the approach, and quantifying the amount of information that could be manually and automatically transferred.

5.4. Results

This section presents the system architecting process developed for both the non-MBSE and MBSE approaches, describes the RBSs and DSMs generated for each approach, and compares the quantities of information manually and automatically transferred between tasks in the DSMs of each approach. Following the comparison, a proposed extensive MBSE approach is described, along with its potential improvements to the original MBSE approach.

5.4.1. Robotic Space System Architecting Process

The high-level, general architecting process developed is shown in Figure 27. The process starts at the Module Level (L4), and proceeds to define the COM module of CCRS and its external interfaces within the Structure Perspective, the data generated and used by the COM within the Data Perspective, the activities the COM performs in the Behavior Perspective, and the functional and non-functional requirements the COM must meet. After the COM level of the architecture is defined, the process repeats to define the COM at the Subsystem and Assembly levels (L5 and L6).

		Perspective			
		Structure	Data	Behavior	Requirements
Organization Level	Module Level (L4)	1	2	3	4
	Subsystem Level (L5)	5	6	7	8
	Assembly Level (L6)	9	10	11	12

FIGURE 27. GENERAL ARCHITECTING PROCESS NUMBERED BY ORDER OF OPERATION.

As the architecting process progressed downward through each level, trade studies were carried out to synthesize the next level of system elements. The trade study process used to define the COM architecture is shown in Figure 28. The process starts by first defining the system functions and evaluation criteria to formulate the problem that the trades study addresses in Trade Study Steps 1-2. Next, system knowledge is generated, recorded, evaluated, and assessed for cross-compatibility through a series of brainstorming activities, research into previous system concepts and available technologies, and a combining of ideas in Trade Study Steps 3-6. Finally, any new concepts are generated, a prioritized list of concepts are recommended, and a prototypes are developed in Trade Study Steps 7-9.

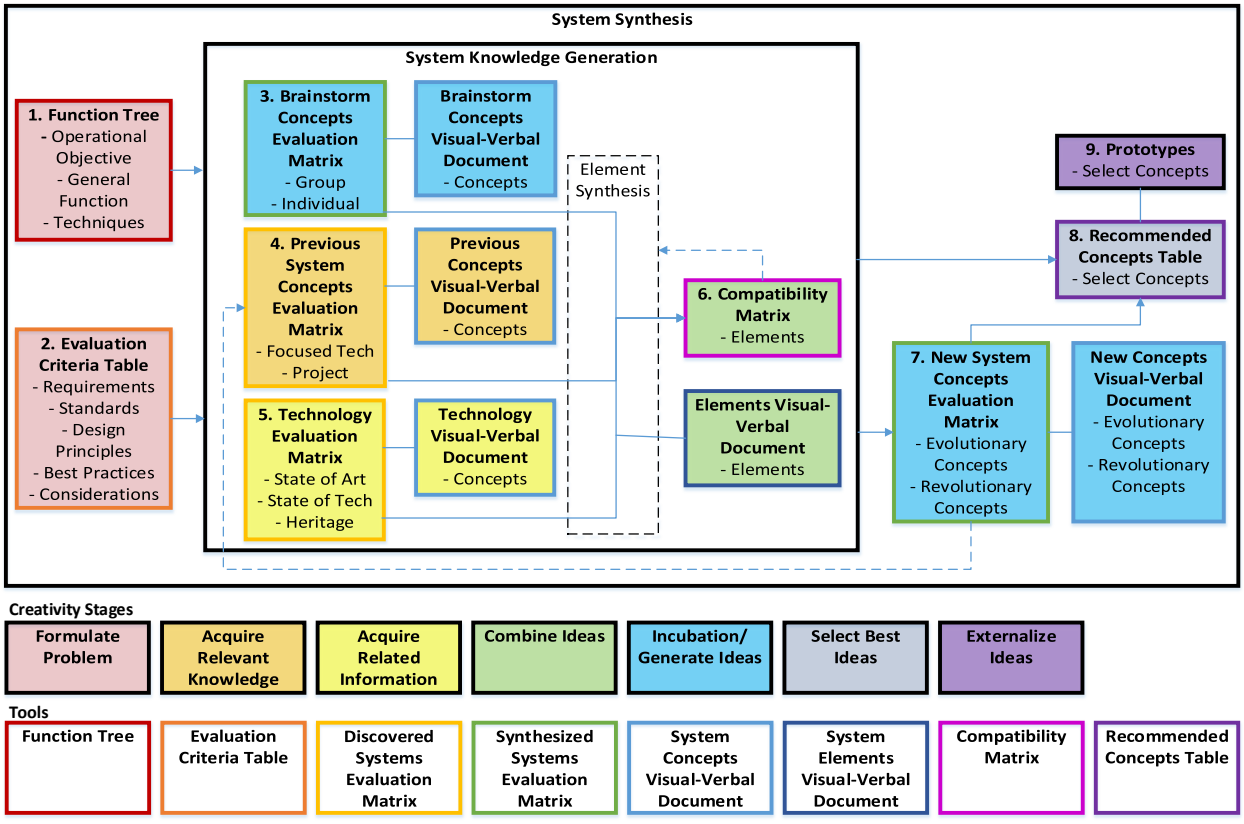


FIGURE 28. OVERVIEW OF TRADE STUDY PROCESS USED TO PROGRESS DOWNWARD THROUGH EACH SYSTEM LEVEL [128].

The trade study process was defined by JPL using a toolkit derived from systems engineering tools and developed based on principles from creativity research, educational psychology, and cognitive psychology [128]. The trade study process defines the structural elements of the next level, as well as captures the information associated with design decisions made in the trade study. Following each trade study, peer reviews were held to review the trades and recommended design concepts, gather technical feedback, and seek approval to proceed down to the next level of architecture definition.

The overall system architecting process was further specialized for the COM and expanded into a set of 26 activities, as shown in the flowchart in Figure 29. The overall process started with definition of the Level 4 structure, data, behavior, and requirements at the COM level, then proceeded with a COM trade study and peer review. Next, the Level 5 structure, data, behavior, and requirements were defined. Trade studies and peer reviews were carried out for the Capture Mechanism, Orientation Mechanism, and Transfer Mechanism. Finally, the Level 6 structure, data, behavior, and requirements were defined for each of the COM subsystems.

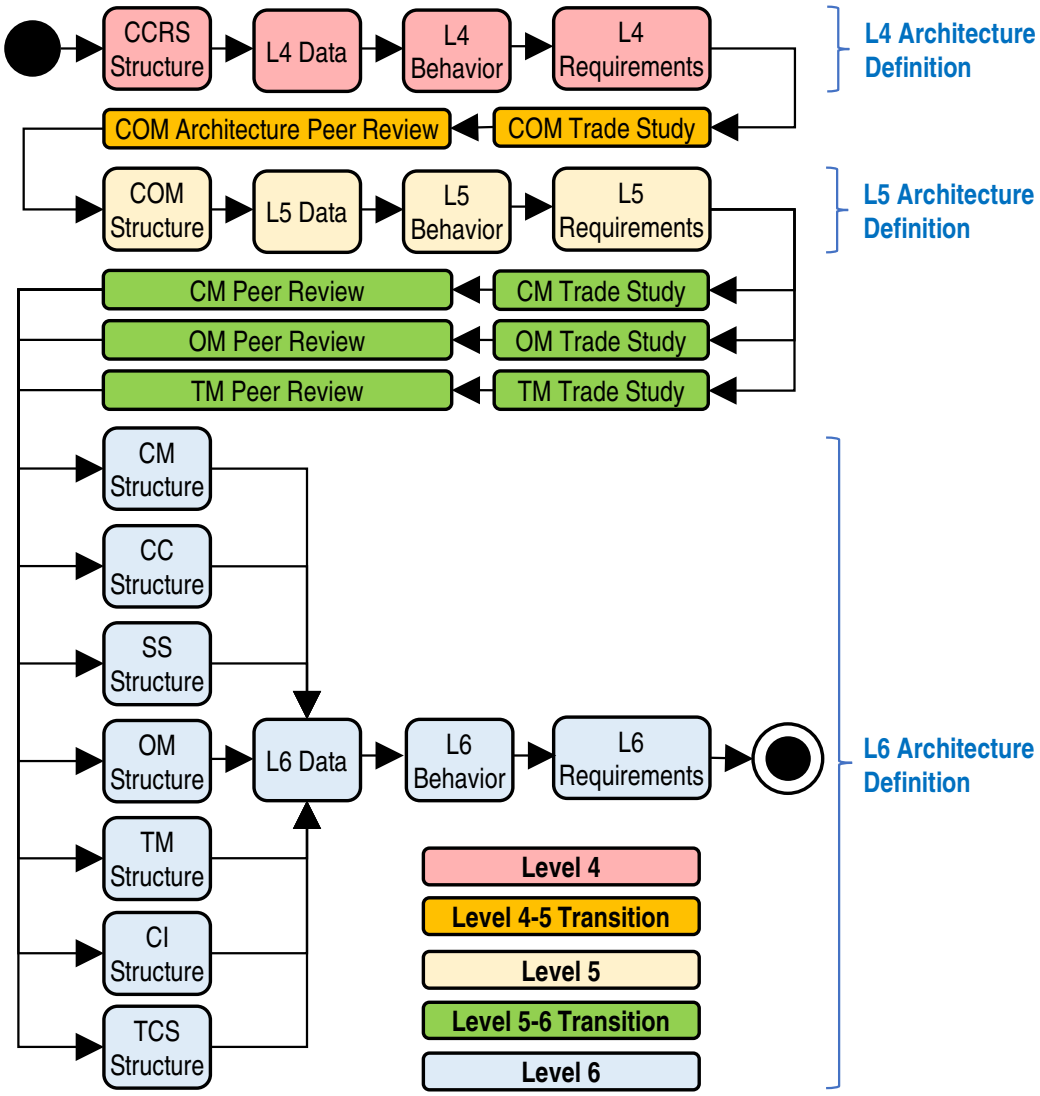


FIGURE 29. FLOWCHART OF THE COM ARCHITECTING PROCESS IMPLEMENTED DOWN TO LEVEL 6 FOLLOWING THE GENERAL ARCHITECTING PROCESS IN FIGURE 27.

The 26 activities in Figure 29 were further decomposed into a unique set of 76 tasks, which are listed in Table 11. The color coding in Table 11 indicates which tasks were associated with the structure, data, behavior, and requirements definition, as well as trade study and peer review activities. The task IDs of the 76 tasks in Table 11 were numbered based on the general order they were performed. These IDs were used as task references in the DSMs.

TABLE 11. LIST OF COM ARCHITECTING PROCESS TASK NAMES AND IDS.

ID	Task
1	Define CCRS Structural Elements
2	Define CCRS Internal Structure
3	Define CCRS Element Specifications
4	Define L4 Data
5	Define L4 Behavior
6	Define L4 Requirements
7	Generate COM Function Tree
8	Generate COM Evaluation Criteria
9	Brainstorm COM Concepts
10	Research Previous COM Concepts
11	Research Relevant COM Technology
12	Assess COM Compatibility
13	Generate New COM Systems
14	Recommend COM Concepts
15	Prototype COM Concepts
16	COM Architecture Peer Review
17	Define COM Structural Elements
18	Define COM Internal Structure
19	Define COM Element Specifications
20	Define L5 Data
21	Define L5 Behavior
22	Define L5 Requirements
23	Generate CM Function Tree
24	Generate CM Evaluation Criteria
25	Brainstorm CM Concepts
26	Research Previous CM Concepts
27	Research Relevant CM Technology
28	Assess CM Compatibility
29	Generate New CM Systems
30	Recommend CM Concepts
31	Prototype CM Concepts
32	CM Peer Review
33	Generate OM Function Tree
34	Generate OM Evaluation Criteria
35	Brainstorm OM Concepts
36	Research Previous OM Concepts
37	Research Relevant OM Technology
38	Assess OM Compatibility
39	Generate New OM Systems
40	Recommend OM Concepts
41	Prototype OM Concepts
42	OM Peer Review
43	Generate TM Function Tree
44	Generate TM Evaluation Criteria
45	Brainstorm TM Concepts
46	Research Previous TM Concepts
47	Research Relevant TM Technology
48	Assess TM Compatibility
49	Generate New TM Systems
50	Recommend TM Concepts
51	Prototype TM Concepts
52	TM Peer Review
53	Define CM Structural Elements
54	Define CM Internal Structure
55	Define CM Element Specifications
56	Define CC Structural Elements
57	Define CC Internal Structure
58	Define CC Element Specifications
59	Define SS Structural Elements
60	Define SS Internal Structure
61	Define SS Element Specifications
62	Define OM Structural Elements
63	Define OM Internal Structure
64	Define OM Element Specifications
65	Define TM Structural Elements
66	Define TM Internal Structure
67	Define TM Element Specifications
68	Define CI Structural Elements
69	Define CI Internal Structure
70	Define CI Element Specifications
71	Define TCS Structural Elements
72	Define TCS Internal Structure
73	Define TCS Element Specifications
74	Define L6 Data
75	Define L6 Behavior
76	Define L6 Requirements

Structure Definition	Requirements Definition
Data Definition	Trade Study
Behavior Definition	Peer Review

5.4.2. Non-MBSE Approach

Table 12 lists the different types of resources identified for the non-MBSE approach, the tools implemented for each resource category, and the total quantity of resources used. The non-MBSE approach captured system architecture information using documents consisting of slides, manuscripts, and spreadsheets. Below is a more detailed description of the key systems engineering artifacts generated in the non-MBSE approach, along with the tools used to generate them:

- Glossary: Excel spreadsheet listing key terms.
- Product Breakdown: PowerPoint slide showing the decomposition of the COM module down to the subsystems and assemblies.
- System Block Diagrams: PowerPoint slides for each module, subsystem and assembly showing the individual elements, along with heater power, temperature sensor, separation device, servo motor control, workhorse motor control, optical, data, sensor power, and mechanical interfaces.
- Product Specifications: Word documents for each module, subsystem, and assembly containing a textual description of the system element, its key attributes, the functions it performs, the requirements it must meet, and other relevant characteristics.
- Data Model: PowerPoint slide showing the specialization of data products used by the COM module, subsystems, and assemblies.
- Scenario: Excel spreadsheet capturing the individual steps that lay out the main operational scenario.
- Requirements: Excel spreadsheet containing the system requirements at the module, subsystem, and assembly levels.

- Function Trees: PowerPoint slides capturing the system operations, objectives, functions, and potential techniques.
- Evaluation Criteria Tables: Excel spreadsheets listing and defining the criteria for the trade studies.
- Evaluation Matrices: Excel spreadsheets used to document, decompose, and evaluate potential system concepts and technologies.
- Visual-Verbal Documents: PowerPoint slides capturing images and descriptions for potential system concepts, technologies, and system elements.
- Compatibility Matrices: Excel spreadsheets evaluation the compatibility of system elements amongst one another.
- Recommended Concept Tables: PowerPoint slides presenting a set of selected concepts for further recommendation and evaluation.
- Peer Review Packages: PowerPoint slides at the module and subsystem levels providing an overview of the system architecture and relevant trade studies.
- Peer Review Advisories: Excel spreadsheets containing technical and programmatic advisories captured during the peer reviews.

TABLE 12. RESOURCES USED WITH THE NON-MBSE ARCHITECTURE APPROACH.

Resource Category	Resource Types	Tool	Qty
Personnel	Systems Engineers, Cognizant Engineers	N/A	8
Slide	Product Breakdowns, System Block Diagrams, Data Models, Function Trees, Visual-Verbal Documents, Peer Review Packages	Microsoft PowerPoint	39
Manuscript	Product Specifications	Microsoft Word	46
Spreadsheet	Scenarios, Requirements, Glossaries, Evaluation Criteria Tables, Evaluation Matrices, Compatibility Matrices, Recommended Concept Tables, Peer Review Advisories	Microsoft Excel	35
Prototype	Prototypes	Physical Model	4
		Total Resources	**132**

Figure 30 shows the DSM generated for the non-MBSE approach with the full set of 76 tasks. The groups of tasks corresponding to the activities depicted in Figure 29 are called out along the diagonal of the matrix for reference. A total of 4,858 information element transfers between tasks were recorded. Since all resources exist as independent documents in the non-MBSE approach, all 4,858 information elements needed to be manually transferred.

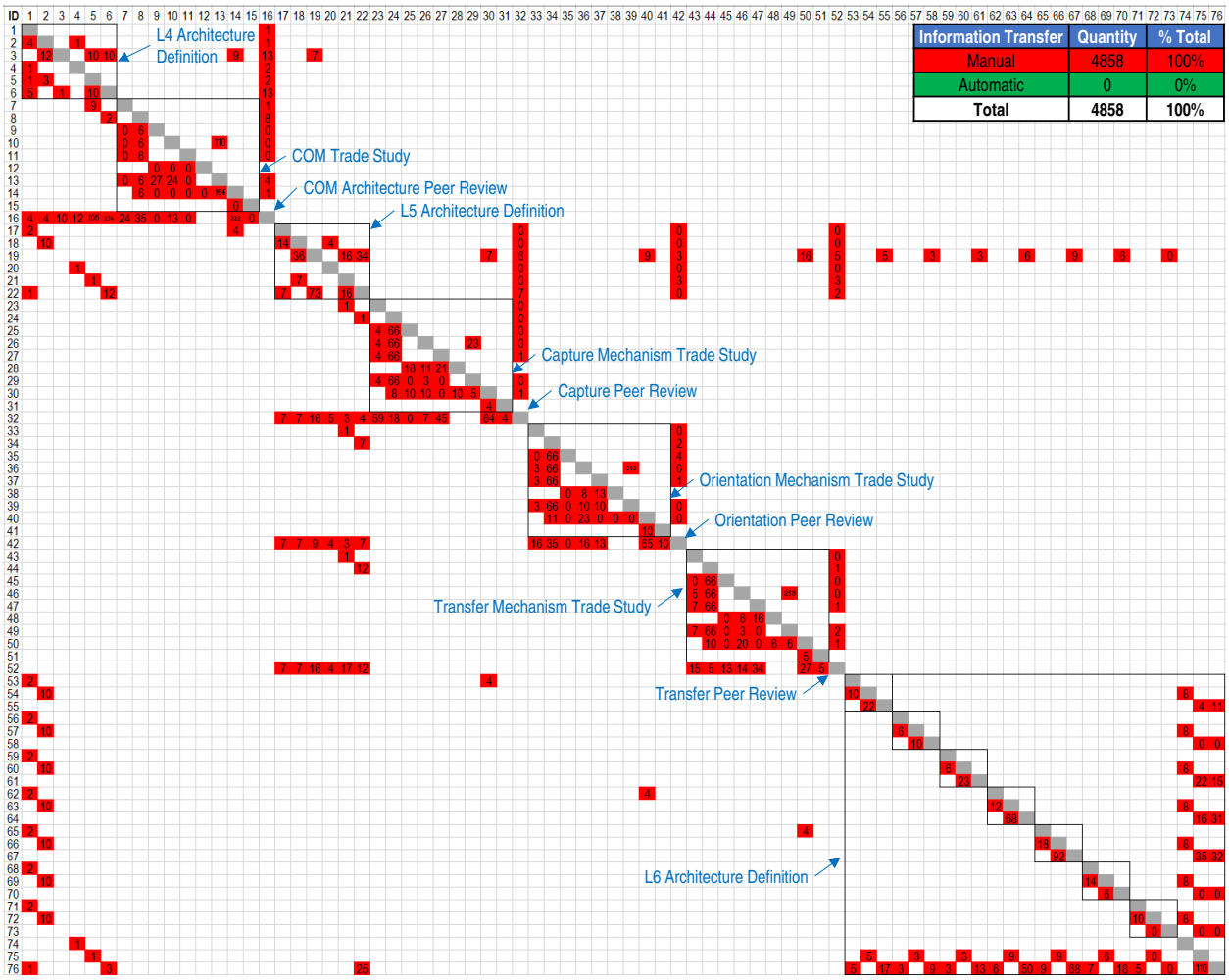


FIGURE 30. DESIGN STRUCTURE MATRIX FOR THE NON-MBSE APPROACH WITH THE GROUPS OF TASKS CORRESPONDING TO THE ACTIVITIES DEPICTED IN FIGURE 29 CALLED OUT ALONG THE DIAGONAL FOR REFERENCE.

5.4.3. MBSE Approach

Table 13 lists the different types of resources identified for the MBSE approach, the tools implemented for each resource category, and the total quantity of resources used. The MBSE approach captured system architecture information using SysML diagrams tied to a single system model generated in Cameo Systems Modeler. The SysML diagrams used in the MBSE approach include Block Definition Diagrams (BDD), Internal Block Diagrams (IBD), Activity Diagrams (AD), and Requirements Diagrams (RD). The Glossary Table available in Cameo Systems

Modeler was also used. Below is a more detailed description of the key systems engineering artifacts generated in the MBSE approach, along with the SysML diagrams used in Cameo Systems Modeler to generate them:

- Glossary: Glossary Table listing key terms.
- Product Breakdown: BDD showing the decomposition of the COM module down to the subsystems and assemblies. Block specifications captured products specifications for each module, subsystem, and assembly.
- System Block Diagrams: IBDs for each module, subsystem and assembly showing the individual elements, along with heater power, temperature sensor, separation device, servo motor control, workhorse motor control, optical, data, sensor power, and mechanical interfaces.
- Data Model: BDD showing the specialization of data products used by the COM module, subsystems, and assemblies.
- Scenario: ADs capturing the individual actions and control flows that lay out the main operational scenario.
- Requirements: RD containing the system requirements at the module, subsystem, and assembly levels.

The trade study tools, peer reviews packages, and peer review advisories used the same documents and tools as the non-MBSE approach.

TABLE 13. RESOURCES USED WITH THE MBSE ARCHITECTURE APPROACH.

Resource Category	Resource Types	Tool	Qty
Personnel	Systems Engineers, Cognizant Engineers	N/A	8
Slide	Function Trees, Visual-Verbal Documents, Peer Review Packages	Microsoft PowerPoint	28
Spreadsheet	Evaluation Criteria Tables, Evaluation Matrices, Compatibility Matrices, Recommended Concept Tables, Peer Review Advisories	Microsoft Excel	32
Prototype	Prototypes	Physical Model	4
Block Definition Diagram	Product Breakdowns, Data Models	Cameo Systems Modeler	2
Internal Block Diagram	System Block Diagrams	Cameo Systems Modeler	9
Block	Product Specifications	Cameo Systems Modeler	46
Activity Diagram	Scenarios	Cameo Systems Modeler	27
Requirements Diagram	Requirements	Cameo Systems Modeler	1
Profile	Classifications	Cameo Systems Modeler	1
Glossary Table	Glossaries	Cameo Systems Modeler	1
		Total Resources	**159**

Figure 31 shows the DSM generated for the MBSE approach with the full set of 76 tasks. The same 4,858 elements that were transferred between tasks in the non-MBSE approach were also transferred between tasks in the MBSE approach. The only difference between the non-MBSE and MBSE approaches are the resources that the information is stored in and retrieved from (the MBSE approach utilizes model views and profiles in place of some of the documents used in the non-MBSE approach). Using the MBSE resources, 630 of the information transfers were able to be automated, since the MBSE model allows for explicit links and associations between information elements. Manual information transfer was, therefore, reduced to 4,228 elements. Note that even though an MBSE model was used to capture the COM architecture information, most of the information transfer between tasks were still manual. This is because documents were still used by the personnel to carry out many of the tasks associated with the trade studies and peer reviews.

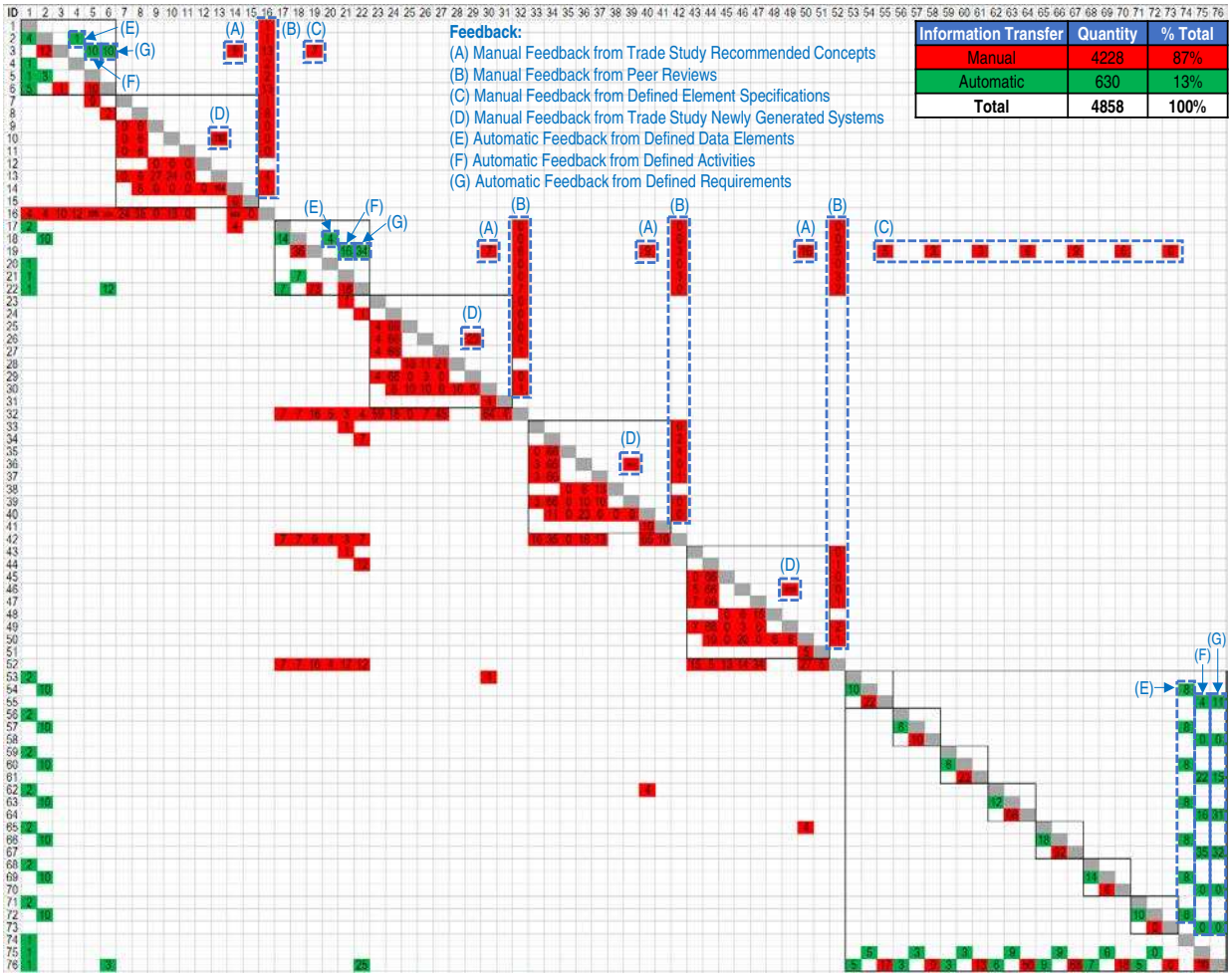


FIGURE 31. DESIGN STRUCTURE MATRIX FOR THE MBSE APPROACH WITH FEEDBACK LOOPS LABELED FOR REFERENCE.

5.4.4. Comparison of the Non-MBSE and MBSE Approaches

Table 14 compares the number of manual and automatic information transfers between tasks for both the non-MBSE and MBSE approaches. The MBSE approach had a smaller number of manual information transfers, and a greater number of automatic information transfers, between tasks than the non-MBSE approach. With the MBSE approach, 13% of the information was able to be associated through SysML relationships in the COM system model, allowing the information to be automatically transferred between tasks.

TABLE 14. NUMBER OF MANUAL AND AUTOMATIC INFORMATION TRANSFERS BETWEEN TASKS FOR THE NON-MBSE, MBSE, AND EXTENSIVE MBSE APPROACHES.

Information Transfer	Non-MBSE Approach		MBSE Approach		Extensive MBSE Approach	
	Quantity	% Total	Quantity	% Total	Quantity	% Total
Manual	4858	100%	4228	87%	931	19%
Automatic	0	0%	630	13%	3927	81%
Total	**4858**	**100%**	**4858**	**100%**	**4858**	**100%**

5.4.5. Extensive MBSE Approach

As reported in Table 14, the MBSE approach used to architect the COM still utilized documents for its trade studies and peer reviews, resulting in 87% of the information transfer still being manually executed. To increase automatic information transfer, opportunities to apply MBSE resources to the trade study and peer review tasks were explored. This included replacing the remaining document-based resources, such as spreadsheets and PowerPoints, with MBSE-based resources, such as Instance Tables, Dependency Matrices, Profiles, and BDDs. Below is a more detailed description of the new systems engineering artifacts proposed for the extensive MBSE approach, along with the SysML diagrams used in Cameo Systems Modeler to generate them:

- Function Trees: BDDs would capture the system operations, objectives, functions, and potential techniques using activity blocks. The BDDs would replace the equivalent PowerPoint slides.
- Concept and Element Visuals: BDDs would capture images and descriptions for potential system concepts, technologies, and system elements using blocks displaying image properties. The BDDs would replace the Visual-Verbal Document PowerPoint slides.
- Concept Compositions: Dependency matrices would be used to decompose system concept. This would replace the system decompositions in the Evaluation Matrix Excel spreadsheets.

- Evaluation Criteria Tables: Instance Tables would list the criteria for the trade studies, using Blocks and Value Types to capture and define trade study evaluation criteria. This would replace the equivalent Excel spreadsheets.
- Evaluation Matrices: Instance Tables would be used to document and evaluate potential system concepts and technologies. This would replace the equivalent Excel spreadsheets.
- Recommended Concepts Tables: Instance Tables would present a set of selected concepts for further recommendation and evaluation. This would replace the equivalent Excel spreadsheets.

Table 15 shows a table of resources for the extensive MBSE approach that utilizes more MBSE resources than the prior MBSE approach. Figure 32 shows the resulting DSM for the extensive MBSE approach, where the yellow cells represent opportunities for manual information transfer to be automated through converting the non-MBSE resources used in those tasks to MBSE resources. When comparing Table 13 to Table 15, the number and types of resources increase with the use of additional, more specialized MBSE diagrams in the extensive MBSE approach. Through utilizing these additional MBSE resources, the percent of automatically transferred information elements is estimated to increase from 13% to 81%, as shown in Table 14.

TABLE 15. RESOURCES FOR THE EXTENSIVE MBSE ARCHITECTURE APPROACH.

Resource Category	Resource Types	Tool	Qty
Personnel	Systems Engineers, Cognizant Engineers	N/A	8
Slide	Function Trees, Peer Review Packages	Microsoft PowerPoint	4
Spreadsheet	Compatibility Matrices, Peer Review Advisories	Microsoft Excel	8
Prototype	Prototypes	Physical Model	4
Block Definition Diagram	Product Breakdowns, Data Models, Function Trees, Evaluation Criteria Definitions, Concept and Element Visuals	Cameo Systems Modeler	30
Internal Block Diagram	System Block Diagrams	Cameo Systems Modeler	9
Block	Product Specifications	Cameo Systems Modeler	46
Activity Diagram	Scenarios	Cameo Systems Modeler	27
Requirements Diagram	Requirements	Cameo Systems Modeler	1
Profile	Classifications, Evaluation Criteria Types	Cameo Systems Modeler	5
Glossary Table	Glossaries	Cameo Systems Modeler	1
Instance Table	Evaluation Criteria Tables, Evaluation Matrices, Recommended Concept Tables	Cameo Systems Modeler	40
Dependency Matrix	Concept Compositions	Cameo Systems Modeler	16
		Total Resources	**199**

FIGURE 32. DESIGN STRUCTURE MATRIX FOR THE EXTENSIVE MBSE APPROACH.

Implementing the above changes proposed for the extensive MBSE approach would require an internal effort to develop new MBSE templates, patterns, and procedures, as well as education to familiarize additional team members and peers on SysML, which is the primary reason the extensive MBSE approach was not utilized for the initial MBSE approach in this research. However, this effort appears feasible to implement in the future given proper investments in time and resources.

5.5. Discussion

To date, many of the comparisons of MBSE to default systems engineering processes have been performed qualitatively. The value proposition for MBSE is asserted to include improvements in quality, velocity/agility, user experience, and knowledge transfer. These categories are based on a framework developed by McDermott et al. for defining and categorizing MBSE benefits and metrics [129]. Because this study performed a direct, comparative analysis of an MBSE and non-MBSE approach, the results lead to a unique elucidation of the quantitative MBSE benefit categories around MBSE-implementation in practice. A summary of the improvements of the MBSE approach over the Non-MBSE approach for each of the four MBSE benefit categories is shown in Table 16. Additionally, limitations identified with the MBSE approach provide recommendations for future work to improve the approach.

TABLE 16. IMPROVEMENTS OF THE MBSE APPROACH OVER THE NON-MBSE APPROACH FOR EACH OF THE FOUR MBSE BENEFIT CATEGORIES.

MBSE Benefit Categories	Improvements of the MBSE Approach over the Non-MBSE Approach
Quality	Improved quality by reducing risk of defects incurred by miscommunication due to automated knowledge transfer of 630 of 4,858 knowledge elements (13% of knowledge elements).
Velocity/Agility	Velocity benefits of the MBSE approach were not directly apparent. Improved velocity was achieved through reducing work time due to automation in 35 of 76 architecting tasks (46% of tasks), but these benefits were offset by the additional time required to set up the MBSE model.
User Experience	Improved user experience by reducing the burden of systems engineering tasks due to automation in 35 of 76 architecting tasks (46% of tasks).
Knowledge Transfer	Improved knowledge transfer within the COM engineering team in 630 of 4,858 knowledge elements (13% of knowledge elements) due to automated knowledge transfer associated with architecture definition tasks. No improved knowledge transfer between the COM engineering team and external peer reviewers in 1,361 of the 4,858 knowledge elements (28% of knowledge elements) due to no automated knowledge transfer associated with peer review tasks.

5.5.1. Implications for Quantifying the Quality Benefits of Model-based Systems Engineering

Automatic information transfer improves the quality of the system and its associated systems engineering artifacts by reducing the risk of defects incurred by miscommunication from manual information transfer. As quantified in this study of a practical MBSE process, MBSE only automated 13% of the total information transfer during architecting. This study measured that even in MBSE-intensive architecture processes, many of the steps of architecting, review, tradeoff, and information transfer are still performed manually. This type of result suggests that the quality benefits of MBSE may be relatively small until an architecture process can realize a high-level of MBSE-enabled automation, implying only minimal improvement in system quality due to automatic information transfer in MBSE efforts that are isolated or incomplete.

5.5.2. Implications for Quantifying the Velocity/Agility Benefits of Model-based Systems Engineering

Work required to otherwise manually transfer information between tasks and check for consistency was slightly reduced through automatic information transfer. The MBSE approach enabled automation of 13% of the total information transfer, affecting 35 of the 76 architecting tasks (46% of the tasks). This implies only minimal improvement in velocity due to only minimal potential reduction in work time from automatic information transfer.

Iteration and rework due to feedback are also major drivers in velocity, particularly since these feedbacks are often unplanned and destabilizing [119]. Most of the information feedback in JPL's documented MBSE architecting process were through manual feedbacks from trade study, peer review, and element specification activities, as labeled (A), (B), (C), and (D) in Figure 31. Feedback that occurs the furthest distance above the diagonal in the DSM indicate a greater number of activities that may need to be repeated in an iteration, which can have the largest impact on hindering velocity. Manual feedback from peer review and element specification

activities fell into this category, and the rearchitecting that occurred from this feedback were primary drivers in the length of time required to define the COM architecture. These discussion points illustrate again that as long as MBSE enabled automatic information transfer is excluded from impactful design activities, such as peer review and element specifications, MBSE's impact on architecting velocity will be limited.

Another limitation with the MBSE approach was that a greater investment in time was required to set up and develop the models with the MBSE approach, relative to composing the documents in the non-MBSE approach. This observation is in line with Madni et al., who also recognize that systems engineering initiatives that employ an MBSE approach require greater upfront investing in the earlier stages of the systems life cycle than needed with traditional systems engineering [26]. Therefore, velocity benefits of the MBSE approach were not directly apparent, as the minor velocity benefits from the automatic information transfer during the architecting process were also offset by the additional time required to set up the MBSE model. This initial time investment associated with model setup should be considered when assessing the overall velocity benefits of an MBSE approach during system architecting.

5.5.3. Implications for Quantifying the User Experience Benefits of Model-based Systems Engineering

User experience was improved through reduced burden of systems engineering tasks and support for automation. The MBSE approach showed reduced burden of manual information transfer through automation for the Level 4, Level 5, and Level 6 architecture definition tasks, but not for tasks associated with the trade studies and peer reviews. The architecture definition tasks consisted of 36 of the 76 tasks (47% of the tasks), and the trade study and peer reviews consisted of 40 of the 76 tasks (53% of the tasks). The MBSE approach automated information

transfer only in the architecture definition tasks, in which 35 of the 36 architecture definition tasks were automated (or 46% of the total number of architecting tasks). This can be attributed to the fact that the MBSE tool and language utilized in the MBSE approach did not have well-defined information constructs, diagrams, and templates for the trade study and peer review tasks. Therefore, the engineering team found it a challenge to implement ideation, design tradeoff studies, and reviews with MBSE using Cameo Systems Modeler and SysML. This is in line with a survey performed by Huldt and Stenius, which indicated greater value with MBSE in architecting and design tasks, over decision support tasks and technical reviews [46].

5.5.4. Implications for Quantifying the Knowledge Transfer Benefits of Model-based Systems Engineering

For the architecting processes studied here, automatic information transfer with the MBSE approach only occurred within the architecture definition tasks, and only accounted for 13% of the total information transfer during architecting. These automations aided with transfer of system knowledge between the Cognizant Engineers and Systems Engineers, as well as collaborative efforts between engineering team members associated with these tasks. Since automatic information transfer was not executed in the peer reviews, the MBSE approach did not offer knowledge transfer benefits between the engineering team and the external peer reviewers. Peer reviews contributed to 4 of the 76 tasks (5% of the tasks) and 1,361 of the 4,858 knowledge element transfers (28% of the knowledge element transfers). MBSE was not implemented for peer reviews due to the current limitations of NASA's MBSE tool, process, and language to generate all the desired views for the peer review presentations, the current lack of a method to collect reviewer feedback and integrate them with the system model, and unfamiliarity of the reviewers with the SysML language. Carlson and Vaneman similarly found in a survey that only

a small percent of Preliminary Design Review (PDR) questions could be addressed with current MBSE methods, and highlighted the need to develop new visualizations for technical reviews to adequately address these needs [130].

5.5.5. Future Work

For future work, these findings suggest a series of follow-ups and advancements, including testing the methodology on additional systems, testing the methodology with different MBSE languages and tools, testing the methodology with two independent development teams, expanding the approach to include parametrics, and developing and testing the extensive MBSE approach with increased automatic information transfer:

- Testing the methodology on additional systems: Applying the MBSE and non-MBSE approaches on additional systems can help validate the methodology across additional engineering domains, as well as gather additional data on how much information transfer can be automated through applying MBSE methods.
- Testing the methodology with different MBSE languages and tools: Different languages possess unique information construct and diagrams. Different tools possess unique abilities to utilize the language, interact with the system model, and generate templates. These unique capabilities could be applied to the trade study and peer review tasks, and potentially improve the MBSE approach's ability to automate information transfer within these tasks.
- Testing the methodology with two independent development teams: This research was carried out by a single development team, which applied the MBSE and non-MBSE approaches in parallel. With this approach, there is a potential that decisions made for one of the implementation approaches could have influenced the other. Testing the methodology with two

independent development teams could help insure that the decision-making in the MBSE and non-MBSE approaches remain independent and decoupled.

- Expanding the MBSE approach to include parametrics: Mathematical relationships between value properties can be captured within a system architecture description. The MBSE architecture approach used in this research captured block value properties, but did not explicitly model mathematical relationships. Integrating parametric diagrams into the architecture framework and architecting process can provide another means to leverage additional capabilities of MBSE and provide further opportunities for automatic information transfer.
- Developing and testing the extensive MBSE approach with increased automatic information transfer: The current MBSE approach used to architect the COM only utilized MBSE during the architecture definition tasks. The trade study and peer review tasks did not utilize MBSE and were still document-based. To address this issue, potential opportunities to apply MBSE methods to the trade study and peer review tasks were explored. This extensive MBSE approach should be implemented on future system architecting activities to validate the approach and measure its ability to increase automatic information transfer for the trade study and peer review tasks.

In developing this study and these recommendations, the authors acknowledge that MBSE as a discipline is under continuous development. With advancements in the language, tool, and processes of MBSE will come improvements in the as-measured performance of MBSE architecting projects.

5.6. Conclusions

MBSE and traditional, document-based systems engineering (non-MBSE) approaches were applied in parallel to architect an orbiting sample COM system concept for a CCRS payload

concept for the potential MSR campaign. The approaches were applied at three architecture levels of the COM: the module level (Level 4), the subsystem level (Level 5), and the assembly level (Level 6). The approaches also covered trades study and peer review tasks between each architecture level.

To explore the advantages of the MBSE approach, resource breakdown structures for the architecting approaches were generated, a system architecting process was synthesized, architecting process task interactions between resources were mapped out in activity diagrams, quantities of manual and automatic information transfer between tasks were recorded in DSMs, and quantities of manual and automatic information transfer were compared for both the non-MBSE and MBSE architecting approaches. A total of 132 resources were used in the non-MBSE approach, and 159 resources in the MBSE approach. The architecting process was broken down into 76 steps. A total of 4,858 information element transfers were recorded between the various process steps. All 100% of these information elements were transferred manually in the non-MBSE approach. The MBSE approach, on the other hand, was able to automate 13% of these information transfers. Additionally, an extensive MBSE approach that further utilizes MBSE resources for the trade study and peer review tasks was predicted to further increase automation to 81%.

Through performing a side-by-side comparison of the MBSE approach with the non-MBSE approach to architect the COM, several findings from this case study were made:

- The MBSE approach developed for architecting the COM proved effective in establishing the architecture of a robotic space system, which included definition of the structure, data, behavior, and requirements of the system at the module, subsystem, and assembly levels.

- The methodology using the DSMs proved a useful tool to identify the information transferred between tasks during the architecting process and facilitate in quantitatively measuring the benefits of the MBSE approach relative to the non-MBSE approach in terms of automatic information transfer.
- The MBSE approach used to architect the COM provided only minor benefits, with an increased automation of information transfer of only 13% of total information element transfer relative to the non-MBSE approach. Increased automatic information transfer provided potential improvements in system quality, user experience in the architecting approach, and knowledge transfer within the engineering team. Velocity benefits of the MBSE approach were not directly apparent, as the minor velocity benefits from the automated knowledge transfer during the architecting process were also offset by the additional time required to set up the MBSE model.
- A large part of the architecting process, particularly tasks related to trade studies and peer reviews, did not utilize MBSE and still relied on manual information transfer. If MBSE resources were applied to trade study and peer review tasks, the value of MBSE during system architecting could be much higher, potentially up to 81%.
- The conclusions drawn from this study were limited to a single system within the robotic space systems domain, using SysML and Cameo Systems Modeler as the modeling language and tool. Additionally, the MBSE approach did not utilize MBSE resources for all system architecting tasks, which limited the ability to assess the full potential of MBSE during the architecting process. Directions for future work should include testing the methodology on additional systems to validate the methodology across additional engineering domains, testing the methodology with different MBSE languages and tools

to potentially improve the MBSE approach's ability to automate information transfer, testing the methodology with two independent MBSE and non-MBSE development teams to insure that the decision-making in the two approaches remain independent, expanding the MBSE approach to include parametrics to leverage additional capabilities of MBSE and provide further opportunities for automatic information transfer, and developing and testing the extensive MBSE approach to validate the approach and measures its ability to increase automatic information transfer.

The COM architecting process and automatic information transfer quantities developed to assess the implementation effort improvements of the COM architecting process using MBSE, which is the goal of Research Question 2, can now be used to assess the evaluation efficiency improvements of the COM modeling and simulation process using MBSE, which is the goal of Research Question 3. Chapter 6 uses the COM system architecture framework from the COM architecting process to develop a simulation-centric V-model, which is built upon four system representations that participate in the COM architecture modeling and simulation process. The manual and automatic information transfer quantities are used to quantity the amount of knowledge processed manually and automatically for each of the COM system architecture modeling and simulation process activities. The benefits provided by the MBSE modeling and simulation approach over the non-MBSE modeling and simulation approach are then assessed based on how much of this knowledge can be processed automatically vs. manually through leveraging MBSE resources.

6. COMPARATIVE ANALYSIS OF MODEL-BASED AND TRADITIONAL SYSTEMS ENGINEERING APPROACHES FOR SIMULATING A ROBOTIC SPACE SYSTEM ARCHITECTURE THROUGH AUTOMATIC KNOWLEDGE PROCESSING[3]

6.1. Introduction

In the year 2029, an Earth Return Orbiter (ERO) notionally plans to autonomously rendezvous with and capture an Orbiting Sample container (OS) holding up to 30 sample tubes filled with Martian rock and soil samples while in orbit, 300 km above the surface of Mars, as part of a potential joint NASA/ESA Mars Sample Return (MSR) campaign [117] (see Figure 33). To accomplish these tasks, a system aboard the ERO would be required to physically capture, orient, and transfer the OS for containment and assembly into an Earth Entry Vehicle (EEV) for eventual return to Earth. A concept for a Capture and Orient Module (COM) was developed at the NASA Jet Propulsion Lab (JPL) to perform these key functions [22].

[3] This chapter was submitted to *Systems Engineering*. The citation is as follows: Younse, P., J. Cameron, and T.H. Bradley, "Comparative Analysis of Model-based and Traditional Systems Engineering Approaches for Simulating a Robotic Space System Architecture through Automatic Knowledge Processing," submitted to *Systems Engineering*, 2021.

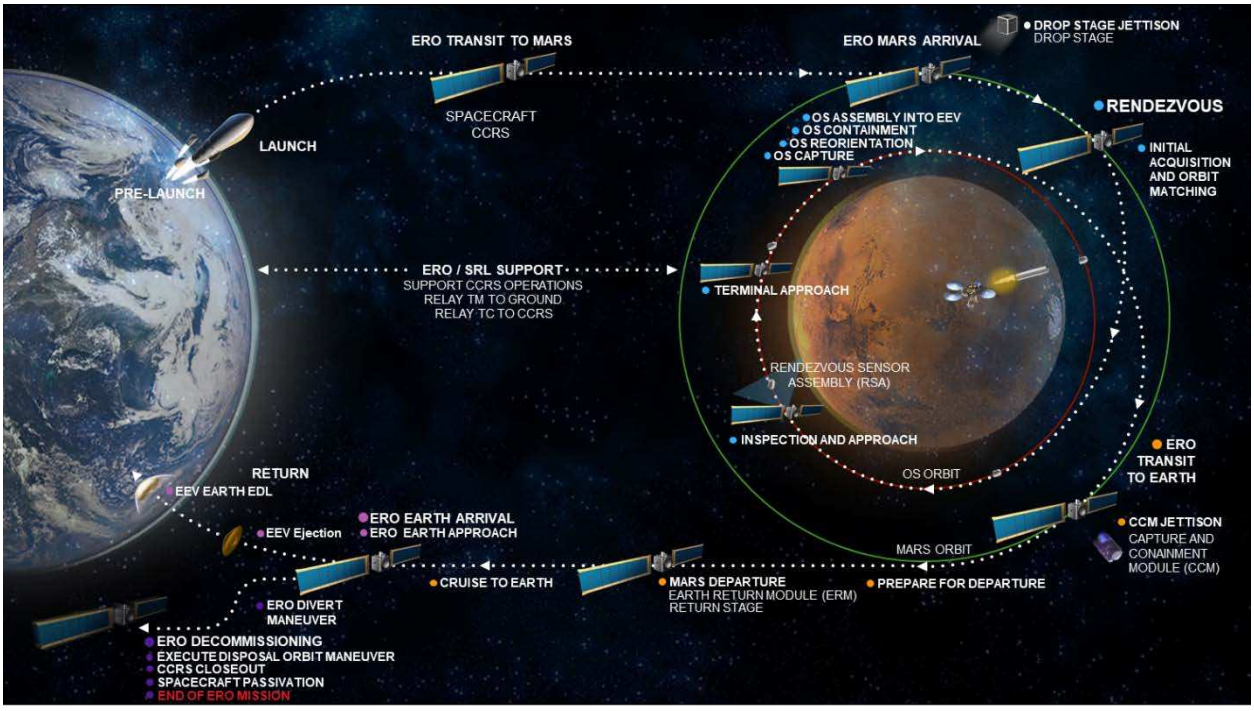


FIGURE 33. NOTIONAL EARTH RETURN ORBITER (ERO) MISSION ARCHITECTURE [22].

The proposed architecture for the COM is complex, composed of 46 unique system elements types, 203 interfaces, and 136 functions, utilizing a total of 4,389 architectural model elements to describe the overall system architecture [131]. Extensive modeling and simulation were performed to verify that the architecture could meet its functional and performance requirements while keeping within system technical constraints. This included simulations to verify OS capture performance, optimize capture trigger sensor configurations, characterize OS-spacecraft interactions, assess vision-based pose-estimation and surface inspection of the OS, estimate operation time, evaluate kinematics of the capture lid and linear transfer mechanism end effector, and estimate power draw and data volume generation [22] [132] [133]. Modeling and simulation were instrumental in developing, characterizing, verifying, and validating the COM architecture, as well as facilitating infusion of key technologies into the final MSR Mission Concept Review (MCR) baseline system architecture.

The National Science Foundation (NSF) reported that there is overwhelming concurrence that simulation in general is key to achieving progress in engineering [134]. Additionally, modeling and simulation represents a core capability needed to address today's complex systems engineering challenges, and lack of modeling and simulation infrastructure leads to knowledge gaps in engineering complex systems [135].

Model-based Systems Engineering (MBSE) is a systems engineering paradigm that can help enable early modeling and simulation of a system, which can provide projects the important opportunity to identify system defects early in the project lifecycle when changes are less expensive to make [136]. Use of MBSE have been demonstrated on space systems. However, there is limited evidence in the literature of case studies that perform a side-by-side comparison of their MBSE approach with a non-MBSE approach using metrics to provide quantitative evidence of the advantages MBSE approaches over traditional, document-based approaches [24]. Henderson and Salado also reported a lack of empirical evidence present in the publicly available literature that supports the hypothesis that MBSE is beneficial for the development of engineered systems [137].

The purpose of this research is to investigate the benefits of applying an MBSE approach to perform modeling and simulation for robotic space system architecture relative to a traditional, Non-MBSE, document-based systems engineering approach. The COM was used as a case study to demonstrate these benefits. Automated knowledge processing within the modeling and simulation process was used as a metric to measure the value of an MBSE approach relative to a Non-MBSE. This research aims to contribute to the literature through providing:

- A case study that compares an MBSE approach side-by-side with a non-MBSE approach for modeling and simulating a robotic space system architecture,

- Quantitative evidence of MBSE's advantages over traditional, non-MBSE approaches for architecture modeling and simulation,
- A modeling and simulation-centric interpretation of the canonical systems engineering V-model that incorporates modeling and simulation elements and activities into the system development lifecycle, enabling comparison of modeling and simulation approaches.

The follow sections proceed to explore the background, methodology, and resulting benefits of the MBSE approach applied in this research. Section II provides background on MBSE, modeling and simulation, the systems engineering V-model, and the COM case study. Section III provides details on the methodology, including the modeling and simulation-centric V-model, the modeling and simulation process, and quantification of the number of knowledge elements manually and automatically processed. Section IV provides a summary of the results, a discussion on the benefits of MBSE, and recommendations for future work. Section V finishes with a conclusion.

6.2. Background

This section provides a brief overview of MBSE, modeling and simulation, the systems engineering V-model, and the COM case study.

6.2.1. Model-based Systems Engineering

MBSE is a systems engineering paradigm that focuses on the use of models to perform systems engineering tasks that are traditionally done using documents [25]. Implementing MBSE requires the use of a modeling language, modeling method, and modeling tool [35]. MBSE modeling languages, methods, and tools offer the possibility to implement rigorous modeling techniques that incorporate traditional systems engineering best practices to develop a central unambiguous, organized, and precise model of the system [52]. Results from a survey performed

by Huldt and Stenius showed MBSE in use across various industries including: aerospace, production, information systems, and automotive [46]. McDermott et al. report on the potential benefits of MBSE across quality, velocity, user experience, and knowledge transfer benefit categories based on responses from participants from a survey of industry, government, and academic organizations [129].

6.2.2. Modeling and Simulation

Modeling and simulation is the discipline that comprises of development and/or use of models and simulations, where a model is a representation of the system, entity, phenomenon, or process, and simulation is a method for implementing the model over time [138]. Models and simulations are indispensable for solving many of today's real-world problems, and are commonly used in design, test and evaluation, decision making, and training [139]. Additionally, virtual system prototypes developed through modeling and simulation that efficiently address important architecture issues can be invaluable in developing and optimizing a system [30].

Recent literature demonstrates the successful use of modeling and simulation within MBSE [140]. This is enabled by successful integration of modeling and simulation into MBSE languages [140] and commercial MBSE tools available to support integrated simulation [141]. Examples of space system projects that utilized MBSE for simulation include the Thirty Meter Telescope [57], Radio Aurora Explorer (RAX) [63], Biomass Mission [142], Iodine Satellite (iSAT) [143], and Laser Interferometer Space Antenna (LISA) mission [144].

Karban et al. investigated the ability to perform system analysis through executable SysML models for requirements verification [57]. They employed an extension to OOSEM called the Executable System Engineering Method (ESEM) for scenario-based power analysis of the Thirty Meter Telescope using the Cameo Simulation Toolkit to verify power requirements of the

telescope. They demonstrated that the method was capable of carrying out the complex analysis needed to show that the system satisfies its peak power requirement.

The RAX CubeSat project aimed to prove the applicability of MBSE for modeling mission operations, focusing on the power and communication subsystems [63]. An executable system model was developed using SysML block definition diagrams, requirement blocks, parametric diagrams, activity diagrams, and state machines. Missions were simulated using Cameo Simulation Toolkit to execute top-level activity diagrams, which in turn called engineering analysis models integrated in ModelCenter to generate time-history energy and data download states. The approach demonstrated the possibility to simulate missions accurately through executable SysML behavioral diagrams.

Gregory et al. analyzed the Payload Data Handling Unit (PDHU) of the Biomass spacecraft to validate spacecraft memory allocation [142]. A high-level functional design of the spacecraft, along with a set of requirements, were modeled in SysML and simulated over a 70-day mission profile to quantify performance in terms of memory usage. The MBSE approach demonstrated flexibility to project changes, with changes automatically propagated through the rest of the model and reflected in subsequent simulation results.

Walker et al. analyzed the iSAT satellite power system to validate spacecraft memory allocation [143]. A closed loop power simulation for a sun and ellipse cycle using activity diagrams and opaque behaviors were modeled in SysML and simulated to calculate power generation, power draw, and battery energy level. The MBSE approach assisted in the preliminary analysis of the design, as well as provided graphical representation of the system and a means for simulation in the same environment.

Charpigny modeled key elements of the Laser Interferometer Space Antenna (LISA) mission to compare an MBSE approach to traditional text-based systems engineering to exchange mission information [144]. Mission structure and behavior was modeled in SysML through a system of nested layers of state machine diagrams and embedded activity diagrams to display spacecraft test-masses potentials and graphically represent science mode states. The MBSE approach demonstrated the ability to combine a descriptive system model with simulations to execute complex system functions, analyze failure modes and associated recovery actions, and promoted exchange of information between the modeler and engineering teams.

The above MBSE case studies assert that there are benefits to MBSE when applied to modeling and simulation of space systems. However, none of these studies perform a side-by-side comparison of their MBSE approach with a non-MBSE approach using metrics to provide quantitative evidence of the advantages MBSE approaches have over traditional, document-based approaches. This research aims to address these gaps in the literature by providing a case study that performs side-by-side comparisons of MBSE and non-MBSE approaches using quantitative metrics. This paper specifically investigates how an MBSE approach can provide user experience benefits with system modeling and simulation in terms of automated knowledge processing.

6.2.3. Systems Engineering V-model

The V-model is a sequential system life cycle model that visualizes systems engineering activities during the life cycle stages [145]. Figure 34 shows a depiction of the systems engineering V-model based on the model presented in the INCOSE Systems Engineering Handbook [145]. The timeline of the system life cycle progresses from left to right. The left side of the V represents the evolving system baseline as the system architecture is decomposed to its

lower level system elements and defined. The right side of the V represents the evolving system baseline as the system elements are integrated and verified.

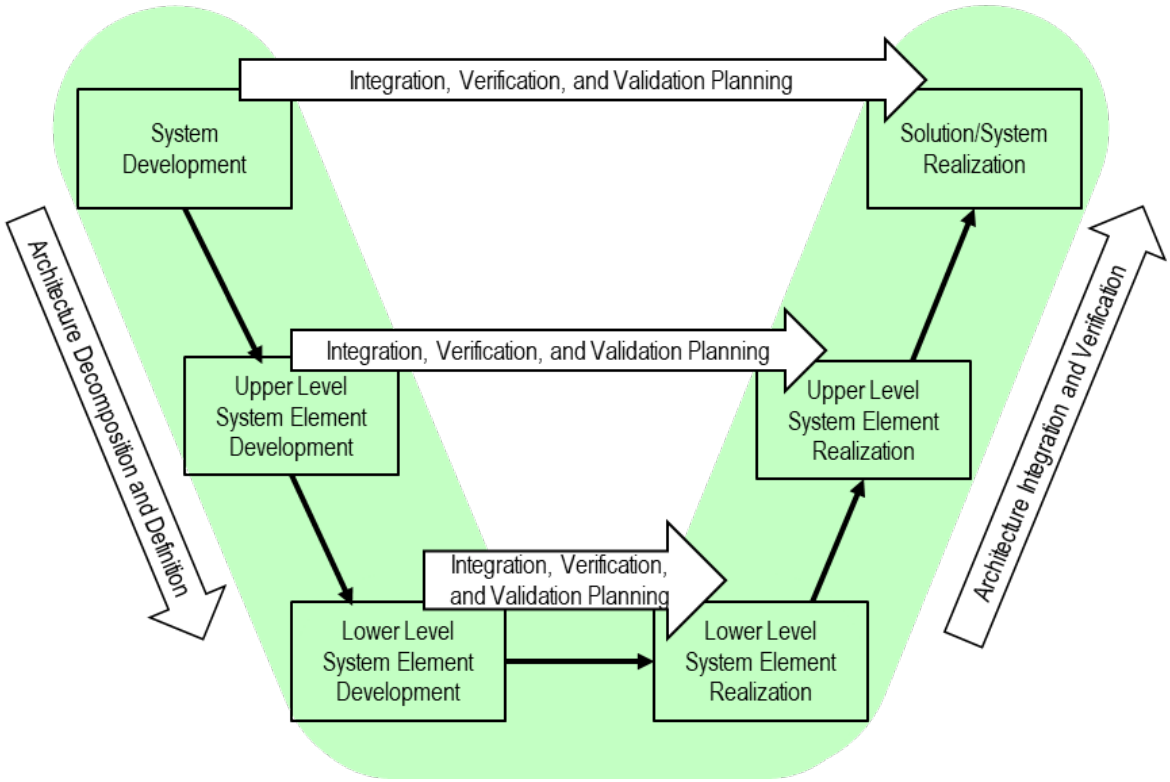


FIGURE 34. GENERAL SYSTEMS ENGINEERING V-MODEL.

Well-established variations on the classical V-model were investigated and evaluated by Graessler et al. [146]. A few characteristic properties of these variations that were seen as potential improvements to the classical V-model model were continuous requirements elicitation and management, model-based design approach, holistic approach for architecture and domain-specific detailing and implementation, in-process status revision, and impulses for enabling digital business models.

Obsbaum et al. [147], and Hatakeyama et al. [148], Eigner et al. [149], and Karban et al. [136] proposed V-models that incorporate modeling and simulation within a model-based engineering environment. The Augmented V-model described by Obsbaum et al. incorporated

modeling and simulation within a parallel bottom-up-branch representing simulation and virtual integration. The MBSE Diamond described by Hatakeyama et al. incorporated modeling and simulation within a mirror image of the physical system V representing a digital twin of the system. The Extended V-model described by Eigner et al. incorporated modeling and simulation into three stages parallel to the left branch of the V: modeling and specification, modeling and first simulation, and discipline-specific modeling and simulation. The JPL V-model described by Karban et al. incorporated modeling and simulation into a central branch down the center of the V representing models at each system level. Even though these V-models incorporate the concepts of modeling and simulation, they do not capture elements and activities of the modeling and simulation process to a level of detail to effectively guide the process within the development life-cycle, as well as provide a means to measure and evaluate the effectiveness of the process. This research will seek to use case study methods to allocate identified elements and activities of a modeling and simulation process to a modeling and simulation-centric viewpoint of the V-model.

6.2.4. Capture and Orient Module Case Study

The COM Pre-Phase A architecture was used as a case study to evaluate the benefits of an MBSE approach over a traditional, non-MBSE approach. The COM is a proposed module within the Capture, Containment, and Return System (CCRS) payload aboard the ERO (see Figure 35). The COM captures, constrains, orients, inspects, and assembles the OS into a Primary Containment Vessel (PCV) in preparation for PCV sealing and installation into an EEV for future delivery to Earth.

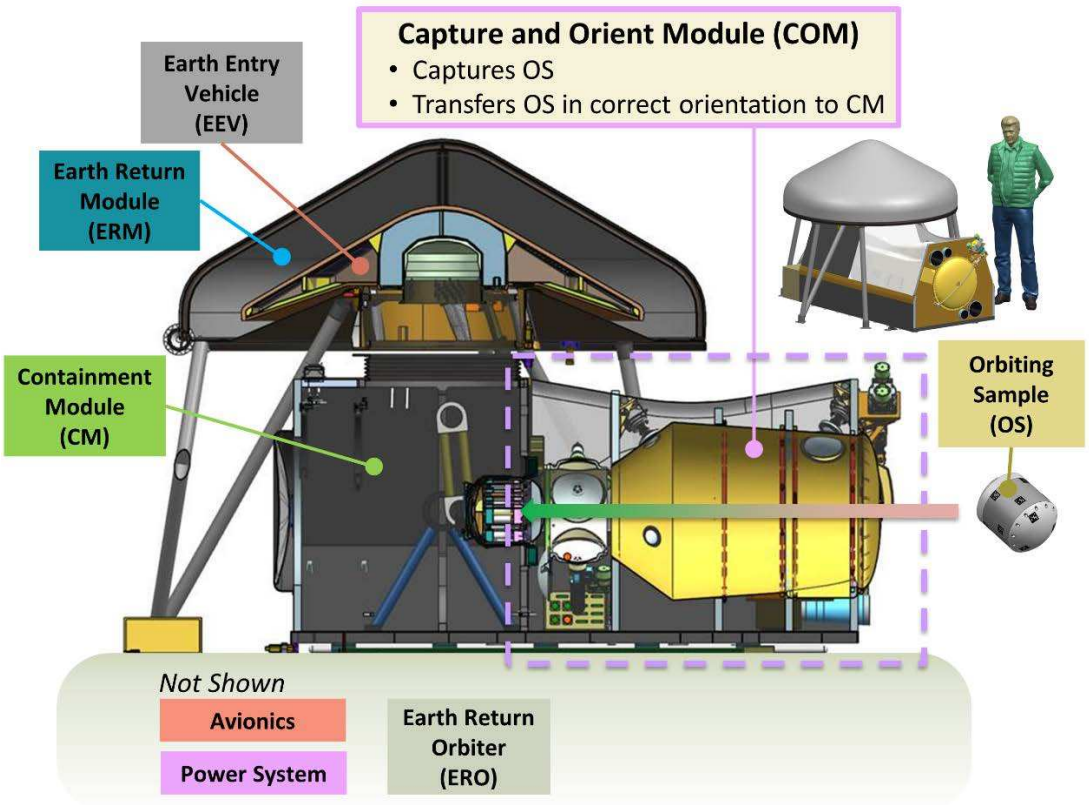


FIGURE 35. NOTIONAL CAPTURE, CONTAINMENT, AND RETURN SYSTEM (CCRS) CONCEPT [22].

Figure 36 shows an early, Pre-Phase A concept of the COM. The COM is organized into three architectural levels: Level 4 (Module Level), Level 5 (Subsystem Level), and Level 6 (Assembly Level). Level 4 represents the overall COM. Level 5 represents the seven major COM subsystems: Capture Mechanism (CM), Sensor System (SS), Capture Cone (CC), Orientation Mechanism (OM), Transfer Mechanism (TM), COM Infrastructure (CI), and Thermal Control System (TCS). The subsystems are further decomposed into assemblies, such as individual actuators, mechanisms, structural elements, and sensors, which are captured in Level 6. Figure 37 illustrates the main operational scenario carried out by the COM from OS capture through assembly into the PCV. Further details on the COM are described by Younse et al. [22].

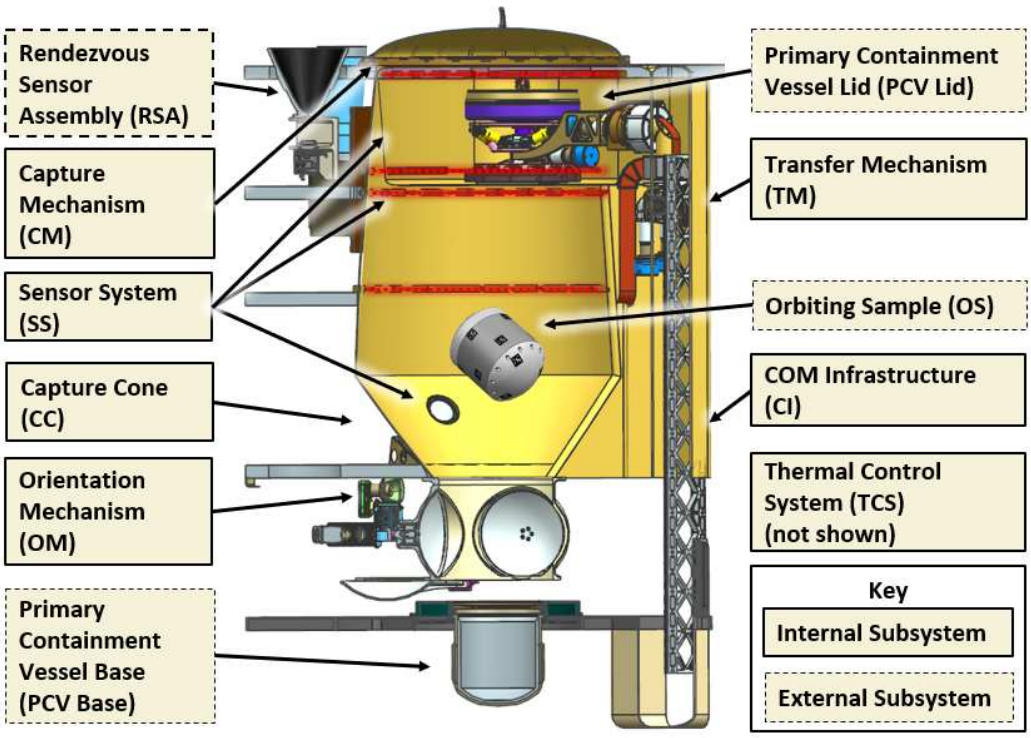


FIGURE 36. NOTIONAL CAPTURE AND ORIENT MODULE (COM) CONCEPT [22].

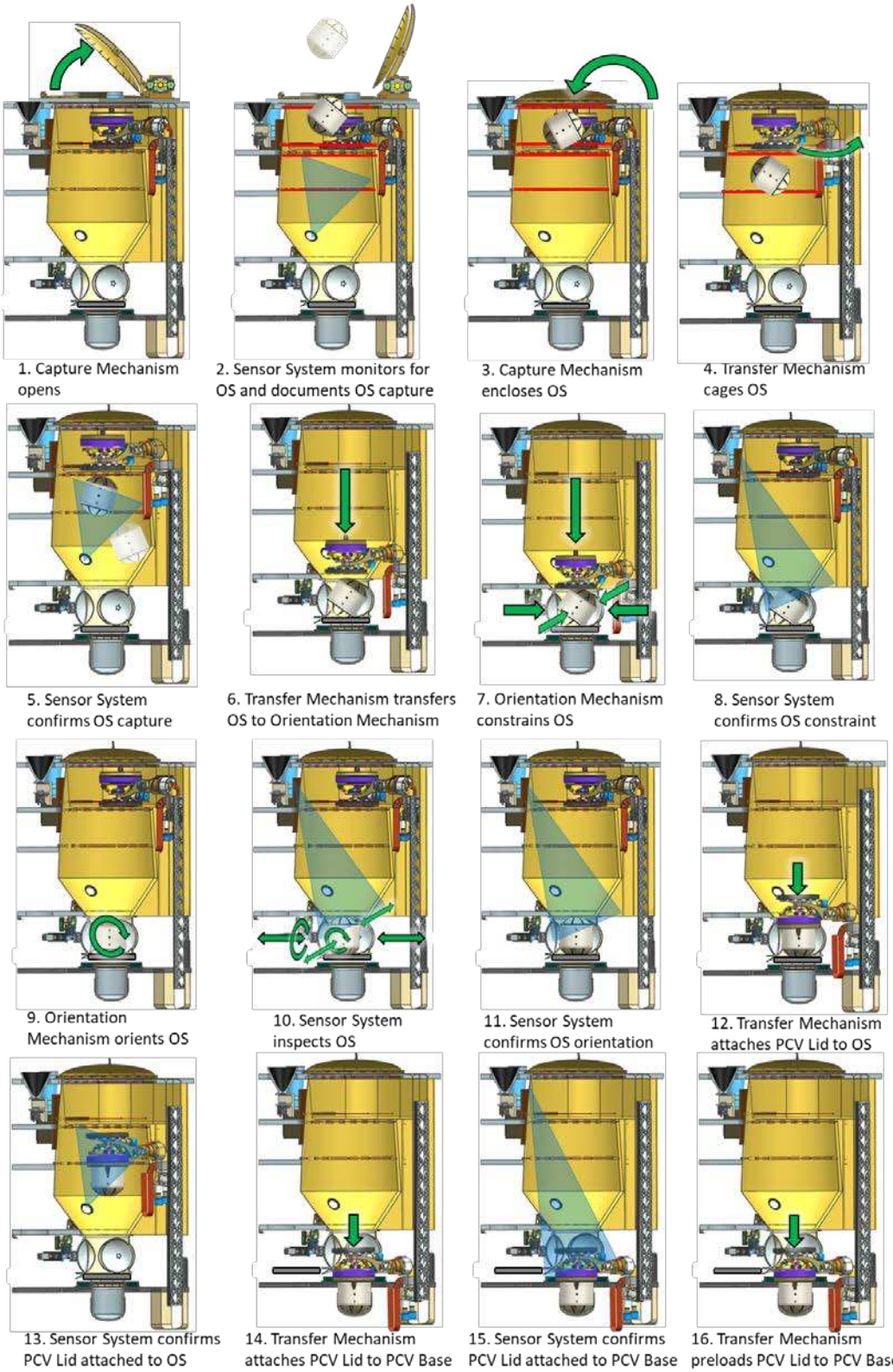


FIGURE 37. NOTIONAL CAPTURE AND ORIENTATION MODULE (COM) OPERATIONAL CONCEPT [22].

The COM architecture was modeled, simulated, and evaluated against two requirements, which included the COM Nominal Load Power and COM Output Data Rate, to compare the MBSE and Non-MBSE modeling and simulation approaches. These requirements address key and driving technical resources of the ERO spacecraft, and were chosen due to their suitability to be verified through modeling and simulating using both the structural and behavioral aspects of the COM architecture.

The COM Nominal Load Power requirement stated that the "COM Nominal Load Power shall not exceed 300 W." The COM nominal load power consists of the total power draw required from all COM subsystems to complete the COM operations, assuming current best estimates for actuator force and torque needs. The 300 W power allocation to the COM was based on available power from ERO to power CCRS systems via the 28 V low voltage bus during CCRS operations.

The COM Output Data Rate requirement stated that the "COM output data rate shall not exceed 200 kbits/s." The COM output data rate consists of the total telemetry and sensing data from all COM subsystem sensors and devices collected during the COM operations desired to be transmitted back to Earth through the Deep Space Network (DSN) in real-time for monitoring operations on the ground. The data rate allocated to the COM was based on minimum data rates available from the ERO communications subsystem assuming a maximum 1.85 AU distance between Earth and Mars during CCRS operations. Initial analyses performed by the CCRS team suggested the need for up to 200 kbits/s of compressed video downlinked in real-time and a similar rate for the images following the video stream, along with other COM related telemetry, to document system operations and observe dynamic events.

The COM architecture was simulated to evaluate and validate these requirements for the overall estimated minimum, average, and maximum operation durations. The timing durations accounted for uncertainties in OS capture time (based on the capture speed driven by the ERO spacecraft guidance, navigation, and control during terminal rendezvous with the OS, occurring during Steps 2 through 4 of Figure 37), degree of rotation required for OS reorientation (based on the initial orientation of the OS when constrained by the Orientation Mechanism, occurring in Steps 7 through 9 of Figure 37), and mechanism angular rotation and linear motion speeds. Nominal load power and output data rates for all COM subsystems and assemblies were assumed consistent for all timing scenarios.

The COM modeling and simulation took place over the course of two years by the COM engineering team at JPL in Pasadena, California, during Pre-Phase A, which is the first of seven phases of the NASA Project Life-Cycle [78]. The MBSE and Non-MBSE modeling and simulation approaches were carried out in parallel over the same course of time by a single team.

6.3. Methodology

A simulation-centric V-model was developed to map out the modeling and simulation activities applied in this research within the context of the system development lifecycle. A three-phase modeling and simulation process consisting of an analysis and modeling phase, a computer programming and implementation phase, and an experimentation phase within the simulation-centric V-model was followed to model, simulation, and verify the requirements of the COM system. This process was carried out in parallel using a both Non-MBSE and an MBSE approach. The number of knowledge elements manually and automatically processed within each approach were quantified and used as metrics to compare the two approaches. Details of the methodology are described further in this section.

6.3.1. Modeling and Simulation-centric V-model

Current V-models do not capture modeling and simulation in enough detail to map out the individual activities involved and quantitatively evaluate different systems engineering design approaches. A new modeling and simulation-centric V-model was synthesized from this case study and several V-model and modeling and simulation process models found in the literature, and used as a tool to compare the Non-MBSE and MBSE approaches evaluated in this research.

The Requirements, Implementation, and Test elements of the V-model proposed by Karban et al. [136], as well as the definition of model elements at each system level were used as the foundation of the modeling and simulation-centric V-model. The central model elements were then expanded to incorporate the Conceptual Model, Computerized Model, and System elements described by Sargent [150], as well as modeling and simulation activities and process flows described by Loper [139] and Sargent [150]. The augmented V-Model containing a parallel bottom-up-branch to account for simulation and virtualization activities described by Obstbaum et al. [147] was added to represent the virtual system, as well as distinguish between physical system testing and virtual system experimentation during the system development cycle.

The general modeling and simulation-centric V-model developed is shown in Figure 38. The proposed V-model is built upon four system representations: the physical system, the conceptual model, the computerized model, and the virtual system. Table 17 provide a description of each of the four system representations. The Physical System is the system representation that follows the requirements and testing branches of the classical V-model. The Conceptual Model and Computerized Model are system representations typically described in modeling and simulation process models. The Virtual System represents an instance of the system generated through simulating the Computerized Model. Since the Computerized Model can be programed and used

to generate multiple instances of a virtual system (e.g., when performing trade studies or sensitivity analyses), it was necessary to explicitly represent the Virtual System in the V-model, as well as the modeling and simulation activities associated with the Virtual System.

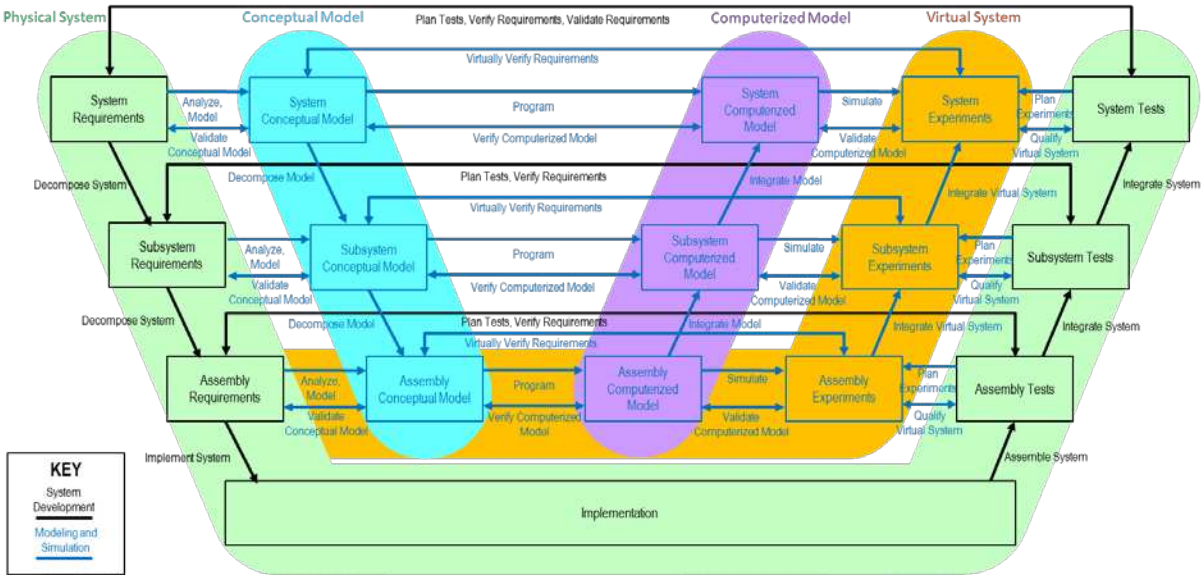


FIGURE 38. GENERAL MODELING AND SIMULATION-CENTRIC V-MODEL.

TABLE 17. V-MODEL SYSTEM REPRESENTATIONS AND DEFINITIONS.

System Representation	Definition
Physical System	The physical system under development following the traditional system development process V.
Conceptual Model	The conceptual representation of the computerized model, including key system elements, capabilities, and assumptions.
Computerized Model	The executable simulation model implemented based on the conceptual model.
Virtual System	The instantiated, simulation of the system generated by the computerized model.

The four system representations are further broken down into a set of V-model elements at each system level, which are described in Table 18. Each the elements are connected through a series of activities, which are implemented as part of the overall system development process. The activities leading down the left-most branch of the Physical System V contribute to the

system definition, decomposition, and implementation, while the activities leading up the right-most branch of the Physical System V contribute to the system assembly, integration, and test. Activities horizontally linking elements along each of the branches of the Physical System V contribute to test planning, verification, and validation. These activities are described in Table 19.

TABLE 18. V-MODEL ELEMENTS AND DEFINITIONS.

V-Model Element	Definition
System, Subsystem, Assembly Requirements	The requirements at the system, subsystem, and assembly levels.
Implementation	The actual procurement, fabrication, or coding of the physical system components.
Assembly, Subsystem, System Tests	The tests run with the physical system at the assembly, subsystem, and system levels.
System, Subsystem, Assembly Conceptual Model	The conceptual models at the system, subsystem, and assembly levels.
Assembly, Subsystem, System Computerized Model	The computerized models at the system, subsystem, and assembly levels.
Assembly, Subsystem, System Experiments	The simulation experiments run with the virtual system at the assembly, subsystem, and system levels.

TABLE 19. V-MODEL PHYSICAL SYSTEM ACTIVITIES AND DEFINITIONS.

Physical System Activity	Definition
Decompose System	Generating the lower level elements, interactions, data, and behaviors of the system, and defining them through a set of requirements.
Implement System	Producing each of the lowest-level system elements from their individual requirements.
Assemble System	Constructing the assembly from the lowest-level system elements.
Integrate System	Constructing the next higher system level from its system elements.
Plan Tests	Designing test procedures to verify and validate that system against its requirements.
Verify Requirements	Checking that the systems meets its specific requirements.
Validate Requirements	Judging that the overall system meets the needs of the customer.

The activities connecting elements within the Conceptual Model, Computerized Model, and Virtual Model are associated with the modeling and simulation process. These activities are

described in Table 20. Similar to the Physical System V, the activities connecting model elements in the branch leading downward are associated with model decomposition, and the activities connecting model elements in the branches leading upward are associated with model integration. The activities leading to the right are associated with modeling and simulation development at each of the system levels. The bidirectional activities are associated with model verification and validation. Like the requirements verification activities carried out via tests on the Physical System, requirements verification is carried out virtually via experiments on the Virtual System, as represented with the activities connecting the elements of the Conceptual Model and Virtual System.

TABLE 20. MODELING AND SIMULATION ACTIVITIES AND DEFINITIONS.

Phase	Modeling and Simulation Activity	Definition
Analysis and Modeling	Analyze	Differentiating out elements of the system requirements and data relevant for the simulation.
	Model	Translating the relevant system elements and data into conceptual model elements.
	Decompose Model	Deconstructing the lower level system elements of the conceptual model from the higher level elements.
	Validate Conceptual Model	Critiquing the conceptual model to ensure that it accurately represents the real-world system to be simulated.
Programming and Implementation	Program	Translate the elements of the conceptual model into executable computerized model elements.
	Integrate Model	Integrating the lower level system elements of the computerized model into the higher level elements.
	Verify Computerized Model	Checking the elements of the computerized model to ensure that it accurately represents the conceptual model.
	Validate Computerized	Critiquing the computerized model to ensure that it produces a virtual system that accurately represents the real-world system to be simulated.
Experimentation	Plan Experiments	Planning the simulation procedure and parameters to execute the experiments.
	Simulate	Executing the experiments, producing instantiations of the system and system elements, executing the system functions, initializing values, and producing experimental results based on the planned experiments.
	Integrate Virtual System	Integrating the lower level system elements of the virtual system into the virtual system higher level elements.
	Qualify Virtual System	Critiquing the virtual system developed through the simulation to ensure that it accurately represents the real-world system to be simulated.
	Virtually Verify Requirements	Checking the experimental results to ensure that they meet the system requirements.

6.3.2. Modeling and Simulation Process

The three-phase modeling and simulation process described by Sargent was followed to carry out the modeling, simulation, and requirements verifications activities for the COM [150]. The modeling and simulation process consisted of: analysis and modeling, programming and implementation, and experimentation. The conceptual model was developed in the analysis and modeling phase. The computerized model was developed in the computer programming and implementation phase. Experiments on the computerized model were executed in the experimentation phase, during which the virtual system was generated and requirements verified from the simulation outputs. Verification and validation of the simulation models, associated data, and simulated output behavior were carried out concurrently, during all phases of development process, as recommended by Sargent [151].

The SysML modeling language and Cameo Systems Modeler tool were utilized for the MBSE approach to allow model integration into the top-level MSR campaign model developed by JPL and ESA [113]. Cameo Systems Modeler and SysML were selected by the MSR campaign due to the extensive experience and resources possessed for the language and tool by both JPL and ESA [114]. Microsoft PowerPoint, Word, and Excel were used to generate the slides, manuscripts, and spreadsheets in non-MBSE approach due to their compatibility with current document-based systems engineering artifacts used at JPL.

6.3.3. Step 1: Analysis and Modeling Phase

The analysis and modeling phase involved analyzing the architecture and requirements of the physical system, modeling the system in the form of a conceptual model, decomposing higher-level model elements, and validation the conceptual model.

The system architecture and requirements of the physical system were manually analyzed at each level to identify knowledge relevant for modeling and simulation needed to verify the COM power and output data rate requirements. This activity was performed at the system (Level 4), subsystem (Level 5), and assembly (Level 6) levels of the COM. This was performed in both the Non-MBSE and MBSE approaches.

The conceptual model of the COM system was manually generated for the non-MBSE approach using a combination of slides, manuscripts, and spreadsheets developed in Microsoft PowerPoint, Microsoft Word, and Microsoft Excel. A slide capturing the product breakdown of the COM was used to document the structural decomposition relationships between Level 4 (Module), Level 5 (Subsystem), and Level 6 (Assembly) elements (see Figure 39). Manuscripts capturing technical specifications for each system element were used to specify element types and attribute values, such as nominal load power and output data rates (see Figure 40). Additionally, mathematical rules that sum up power and data rates from lower level elements were documented within the manuscripts. A spreadsheet capturing the operational concept of the system was used to document control flows, functions, functional decompositions, functional allocations, and functional duration values (see Figure 41). A spreadsheet was used to capture requirements and their associated requirements texts, allocations, and parents (see Figure 42).

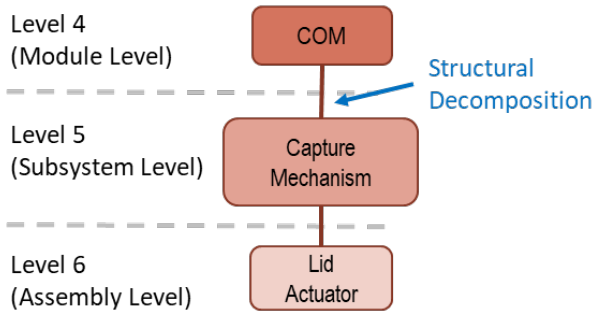


FIGURE 39. EXAMPLE OF THE COM, CAPTURE MECHANISM, AND LID ACTUATOR ELEMENTS FROM THE NON-MBSE PRODUCT BREAKDOWN SLIDE. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

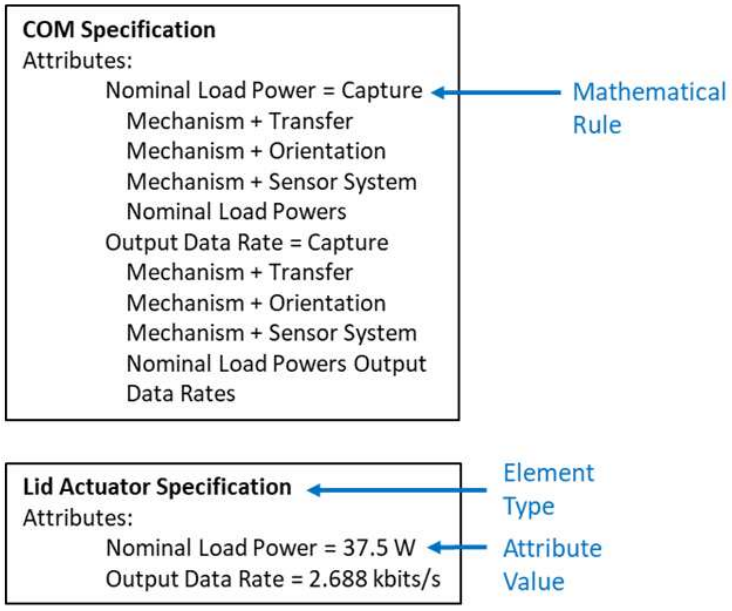


FIGURE 40. EXAMPLE OF COM AND LID ACTUATOR NON-MBSE SPECIFICATION MANUSCRIPTS. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

Scenario Step	Scenario Description L4	Scenario Description L5	Scenario Description L6	Min Execution Time (s)	Avg Execution Time (s)	Max Execution Time (s)
1	COM captures OS	Capture Mechanism opens	Lid Actuator opens Lid	1.3	1.5	1.7
2			Outward Vision System confirms Lid deployment	1.0	1.0	1.0

Control Flow
Functional Allocation
Functional Decomposition
Function
Function Duration Value

FIGURE 41. EXAMPLE OF TWO LEVEL 6 SCENARIO STEPS FROM THE NON-MBSE SCENARIO SHEET. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

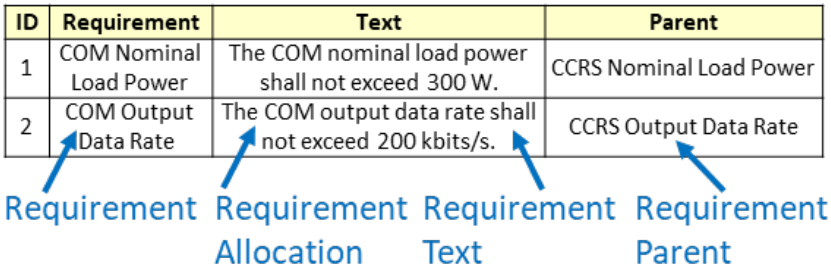


FIGURE 42. THE COM NOMINAL LOAD POWER AND COM OUTPUT DATA RATE REQUIREMENTS REPRESENTED IN THE NON-MBSE REQUIREMENTS SHEET. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

The MBSE approach used a combination of SysML diagrams manually modeled in Cameo Systems Modeler to develop a conceptual model of the COM. A block definition diagram was used to capture element types, structural decomposition relationships between blocks, and attribute values within the bock specifications (see Figure 43). Additionally, mathematical rules were incorporated into the block by applying rollup patterns for both power and data rate attributes using Cameo's built-in Rollup Pattern Wizard (see Figure 44).

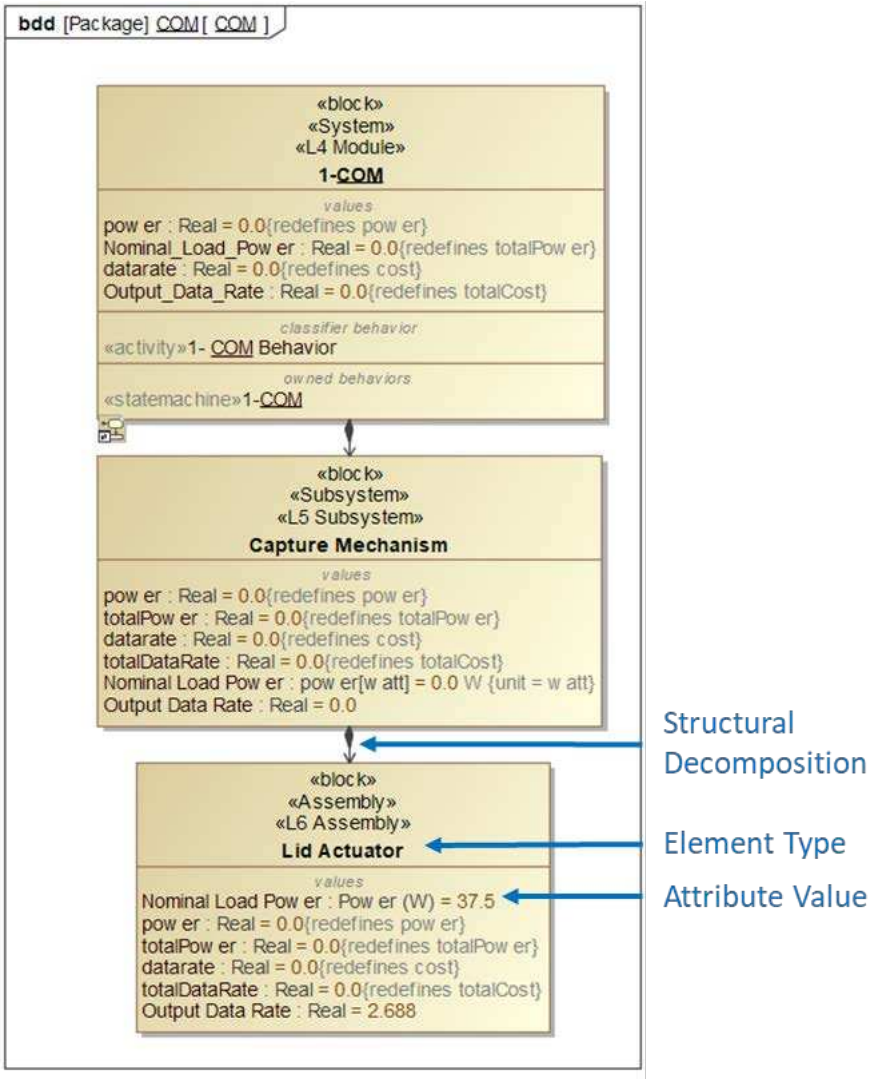


FIGURE 43. EXAMPLE OF THE COM, CAPTURE MECHANISM, AND LID ACTUATOR BLOCKS FROM THE MBSE BLOCK DEFINITION DIAGRAM. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

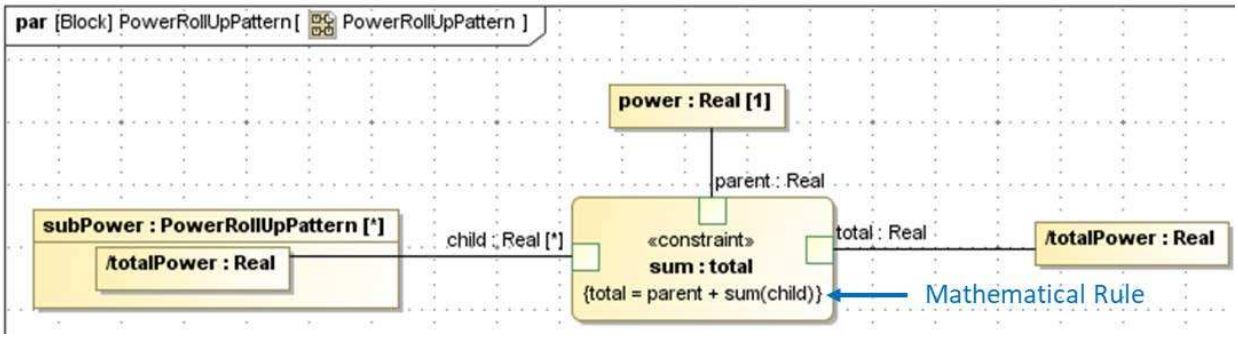


FIGURE 44. MBSE PARAMETRIC DIAGRAM OF THE POWER ROLLUP PATTERN. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

The system behavior was defined for the COM at the top-level using a state machine diagram, and then expanded into an operational concept using a series of activity diagrams at each of the system, subsystem, and assembly levels (see Figure 45). Each activity diagram captured functions, control flows, functional allocations, and functional decompositions. Each of the assembly level actions were further defined using activity diagrams containing the power and data rate read/write pattern shown in Figure 46. During simulation, this pattern reads the nominal load power and output data rate attribute values specified in the element blocks using the "read structural feature" actions, calculates the element's power and data rate for its on and off states using opaque actions, and then writes these values to the element's power and data rate for the duration of the operation using the "add structural feature value" actions. Minimum and maximum function duration values for each assembly level action were captured as duration constraints within the action labeled "operation."

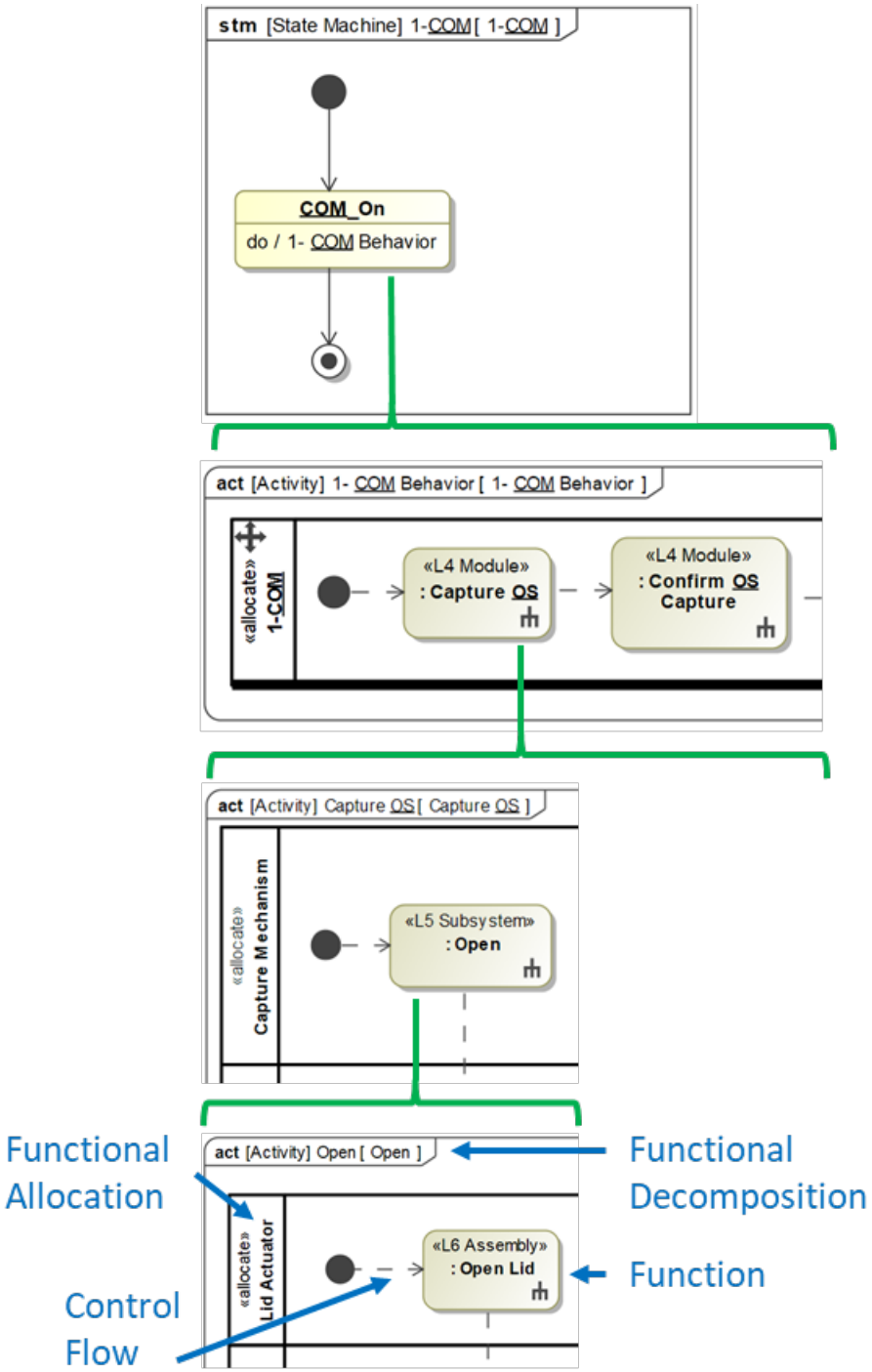


FIGURE 45. MBSE COM BEHAVIOR STATE MACHINE DIAGRAM AND NESTED LEVEL 4, LEVEL 5, AND LEVEL 6 ACTIVITY DIAGRAMS FOR THE OPEN LID ACTION. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

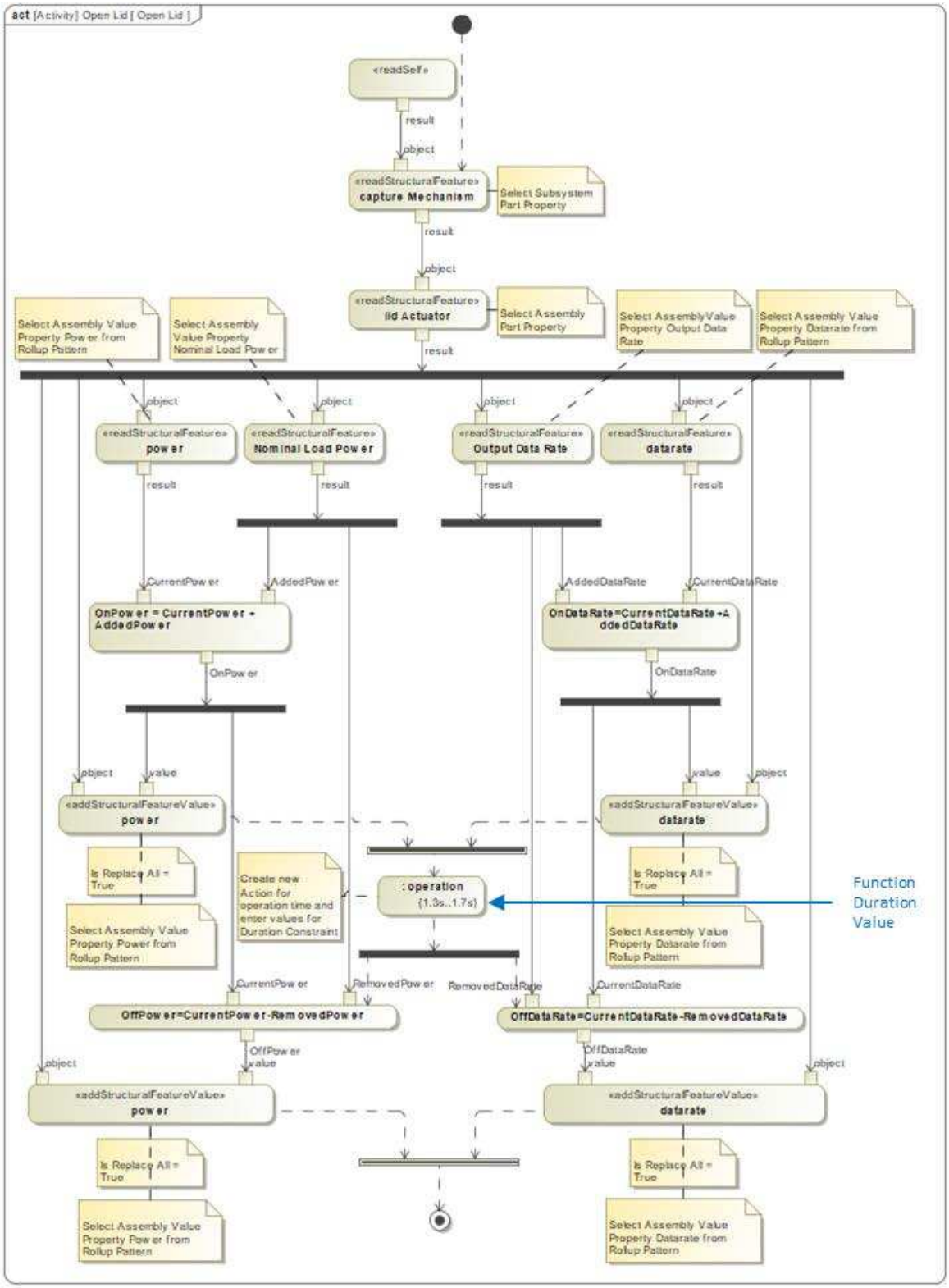


FIGURE 46. MBSE POWER AND DATA RATE READ/WRITE PATTERN USED FOR EACH LOWER-LEVEL ACTION. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

Requirements were captured as requirement blocks within a requirements diagram (see Figure 47). Each block contained the requirement, its parent, the element it was allocated to, and its text. Additionally, Cameo's Requirement Terms Glossary enables the tool to interpret conditional phrases within the requirements text (indicated by the underlined "not exceed" text), along with the numerical values associated with the conditional phrases.

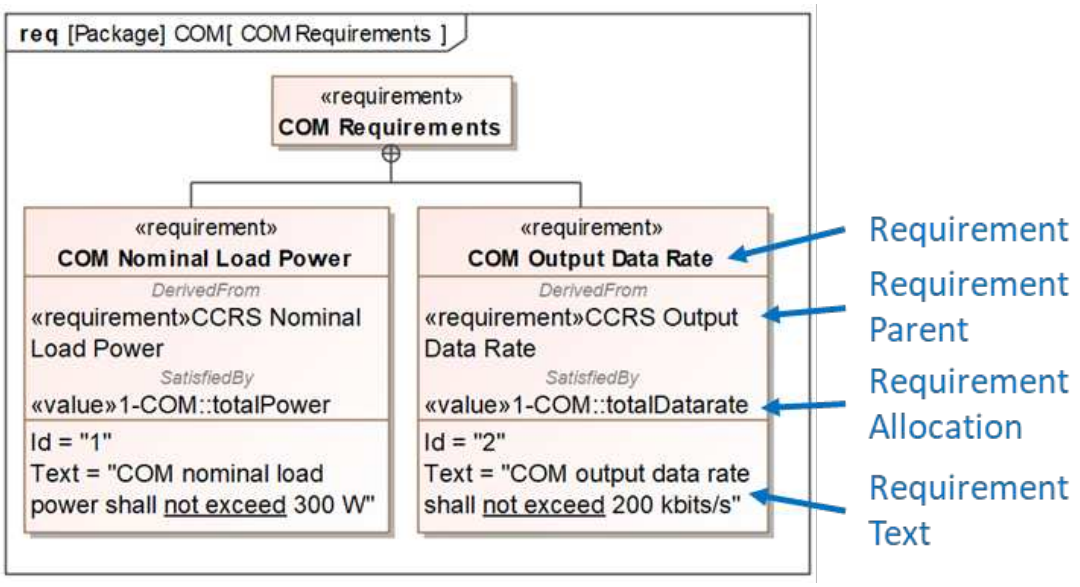


FIGURE 47. THE COM NOMINAL LOAD POWER AND COM OUTPUT DATA RATE REQUIREMENTS REPRESENTED IN THE MBSE REQUIREMENT DIAGRAM. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE CONCEPTUAL MODEL ARE POINTED OUT IN BLUE.

Once the Conceptual Model was complete, the model was manually validated in both the Non-MBSE and MBSE approaches. Conceptual model validation was performed using various techniques, including, face validity, comparison to other models, and animation (only in the MBSE approach). In the face validity technique, the conceptual model diagrams were reviews with individuals on the engineering team and subject matter experts knowledgeable about the system to ensure the model elements, logic, and input-output relationship looked reasonable. In the comparison to other models technique, the diagrams produced in the Non-MBSE and MBSE

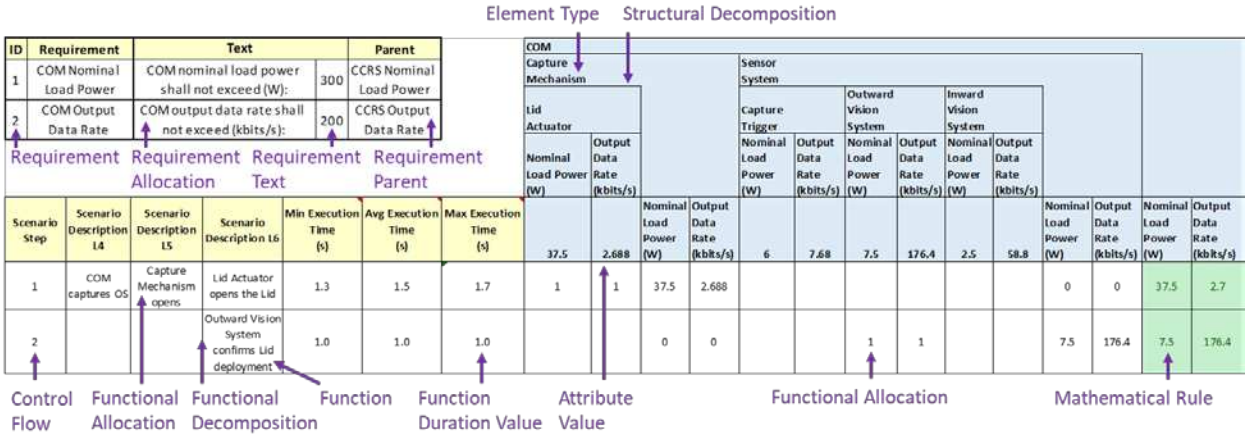
approaches, along with other, previously generated Non-MBSE and MBSE diagrams of the system, were compared and cross-checked with one another. Additionally, the MBSE conceptual model was validated using animation. This involved executing the activity diagrams in Cameo Systems Modeler to automatically step through the sequence of actions to check that the flow is logical and matches the expected behavior of the system. More details on these validation techniques can be found in Sargent [150] and Carson [152].

6.3.4. Step 2: Computer Programming and Implementation Phase

The computer programming and implementation phase involved programing the system computerized model, verifying the computerized model, and validating the computerized model.

A computerized model of the COM system was manually programmed in a spreadsheet using Microsoft Excel for the non-MBSE approach. The spreadsheet pulled system knowledge from the slides, manuscripts, and spreadsheets of the conceptual model, and integrated them into a new spreadsheet, as shown in Figure 48. Element types were copied over to the area of the spreadsheet colored in blue. Rectangular boxes within the blue area were drawn up to represented the structural hierarchy of the system, with lower-level elements represented by smaller rectangular boxes within each of the high-level regions. Nominal Load Power and Output Data Rate values for each assembly were entered into the spreadsheet. Functional Allocations that indicate which scenario steps each of the assemblies are active were indicated by entering a “1” in their corresponding columns. Mathematical Rules that sum up all power and output data rates for each active assembly for each scenario step were entered as formulas in each cell under Nominal Load Power and Output Data Rate columns for each of the subsystems, as well as the overall COM system. In contrast to the Non-MBSE approach described above, the MBSE approach automatically programmed the computerized model from the conceptual model

due to Cameo Systems Modeler's ability to interpret and execute the SysML diagrams. Therefore, no additional manual work was needed to generate the Computerized Model in the MBSE approach.



ID	Requirement	Text		Parent
1	COM Nominal Load Power	COM nominal load power shall not exceed (W):	300	CCRS Nominal Load Power
2	COM Output Data Rate	COM output data rate shall not exceed (kbits/s):	200	CCRS Output Data Rate

Scenario Step	Scenario Description L4	Scenario Description L5	Scenario Description L6	Min Execution Time (s)	Avg Execution Time (s)	Max Execution Time (s)	COM Capture Mechanism lid Actuator Nominal Load Power (W)	COM Capture Mechanism lid Actuator Output Data Rate (kbits/s)	COM Nominal Load Power (W)	COM Output Data Rate (kbits/s)	Sensor System Capture Trigger Nominal Load Power (W)	Sensor System Capture Trigger Output Data Rate (kbits/s)	Sensor System Outward Vision System Nominal Load Power (W)	Sensor System Outward Vision System Output Data Rate (kbits/s)	Sensor System Inward Vision System Nominal Load Power (W)	Sensor System Inward Vision System Output Data Rate (kbits/s)	Nominal Load Power (W)	Output Data Rate (kbits/s)	Nominal Load Power (W)	Output Data Rate (kbits/s)
							37.5	2.688			6	7.68	7.5	176.4	2.5	58.8				
1	COM captures OS	Capture Mechanism opens	Lid Actuator opens the Lid	1.3	1.5	1.7	1	1	37.5	2.688							0	0	37.5	2.7
2			Outward Vision System confirms Lid deployment	1.0	1.0	1.0			0	0			1	1			7.5	176.4	7.5	176.4

FIGURE 48. EXAMPLE OF TWO LEVEL 6 SCENARIO STEPS FROM THE NON-MBSE COMPUTERIZED MODEL. EXAMPLES OF KNOWLEDGE ELEMENTS REPRESENTED IN THE COMPUTERIZED MODEL ARE POINTED OUT IN PURPLE.

In the Non-MBSE approach, the computerized model was manually verified to ensure all necessary elements from the conceptual model were captured and translated correctly into the computerized model. In the MBSE approach, the computerized model was automatically verified by Cameo Systems Modeler, which automatically checked the accuracy, completeness, and correctness of a model, marked invalid elements in the model, and suggested solutions when available.

The computerized model was manually validated in both the Non-MBSE and MBSE approaches. Model validation was performed using various techniques, including walk throughs of the computerized model, operational graphics (looking at power and output data rate profiles in the Time Series Charts to ensure they behaved correctly), and parameter variability-sensitivity

analysis (varying control and noise variables to determine if the directionality of the change in the output were logical).

6.3.5. Step 3: Experimentation Phase

The experimentation phase involved planning experiments, simulating the system, integrating the virtual system across levels, and virtually verifying requirements.

System experiments were planned out manually in both the Non-MBSE and MBSE approaches. This included defining system process parameters relevant to the experiments, experiment scenarios, simulation attributes, and reporting attributes. System process parameters included input, output, noise factor, and control parameters, as depicted in the Taguchi P-Diagram shown in Figure 49. Experiment scenarios were defined based on the noise factor and control parameters in an experimental matrix, as shown in Table 21. Three scenarios were planned: the COM Minimum Execution Time, the COM Average Execution Time, and the COM Maximum Execution Time. Simulation configuration attributes values included start time, duration, step size, and time unit for each experiment scenario. Reporting attributes included axis, grid, trendline, and label properties associated with the time series charts for each experiment scenario. In the Non-MBSE approach, simulation configuration and reporting attributes were incorporated into the spreadsheet and time series charts. In the MBSE approach, simulation configuration and reporting attributes were captured in a dedicated Simulation Configuration Diagram using Simulation Configuration and Time Series Chart blocks, as shown in Figure 50.

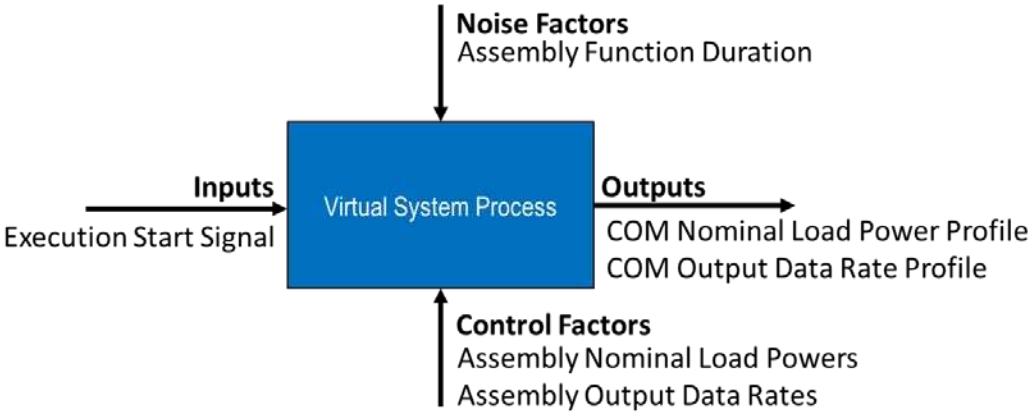


FIGURE 49. GENERAL TAGUCHI P DIAGRAM FOR VIRTUAL SYSTEM EXPERIMENT PLANNING.

TABLE 21. EXPERIMENTAL MATRIX.

Experiment Scenario	Assembly Function Duration	Assembly Nominal Load Powers	Assembly Output Data Rates
COM Minimum Execution Time	Minimum	CBE Power Values	CBE Data Rate Values
COM Average Execution Time	Average	CBE Power Values	CBE Data Rate Values
COM Maximum Execution Time	Maximum	CBE Power Values	CBE Data Rate Values

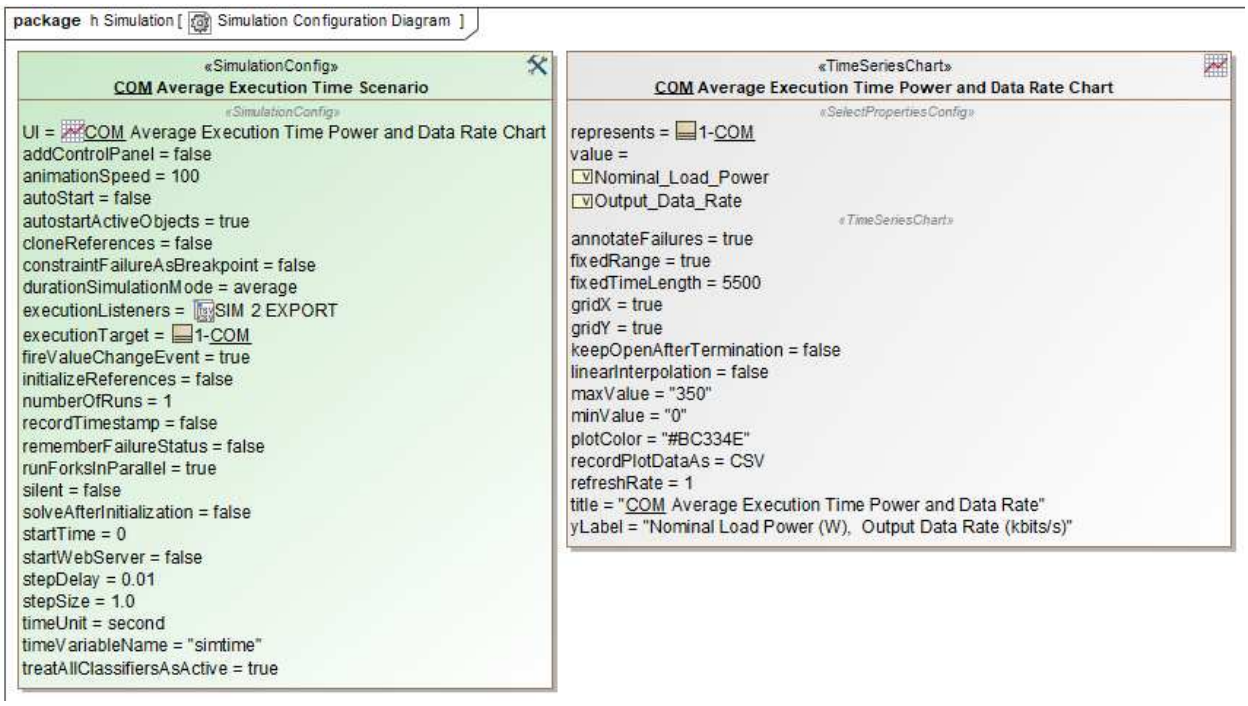


FIGURE 50. EXAMPLE OF A SIMULATION CONFIGURATION AND TIME SERIES CHART BLOCKS IN THE MBSE SIMULATION CONFIGURATION DIAGRAM.

Simulating the system in the Non-MBSE approach involved generating a copy of the computerized model spreadsheet to create a virtual instantiation of the system, entering relevant data into the spreadsheet to initialize values for each experiment, generating a set of results, and plotting the results on time series charts. Since the spreadsheet is a static representation of the system, the output data for a time series set of results needed to be manually generated, formatted, and input into the charts. Figure 51, Figure 52, and Figure 53 show the Non-MBSE approach time series charts displaying the results from the minimum, average, and maximum execution times.

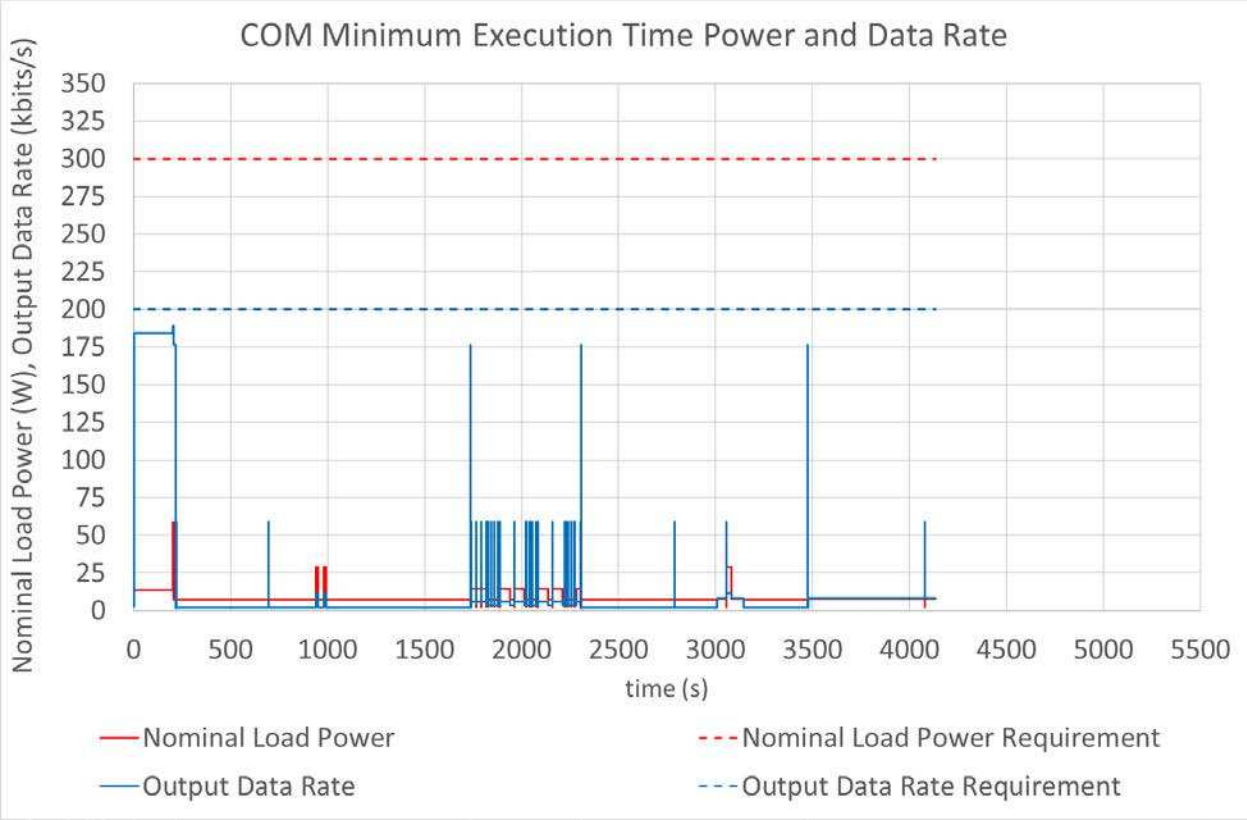


FIGURE 51. NON-MBSE COM MINIMUM EXECUTION TIME POWER AND DATA RATE RESULTS.

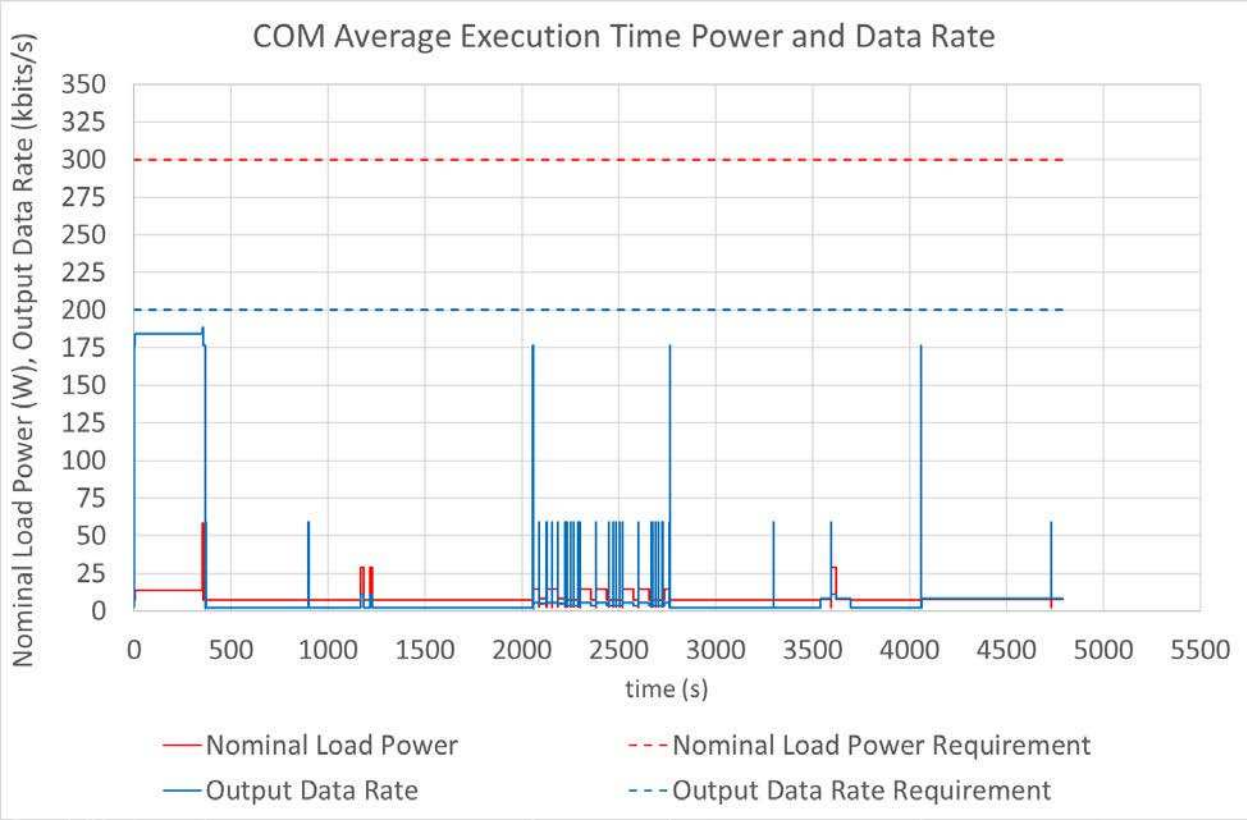


FIGURE 52. NON-MBSE COM AVERAGE EXECUTION TIME POWER AND DATA RATE RESULTS.

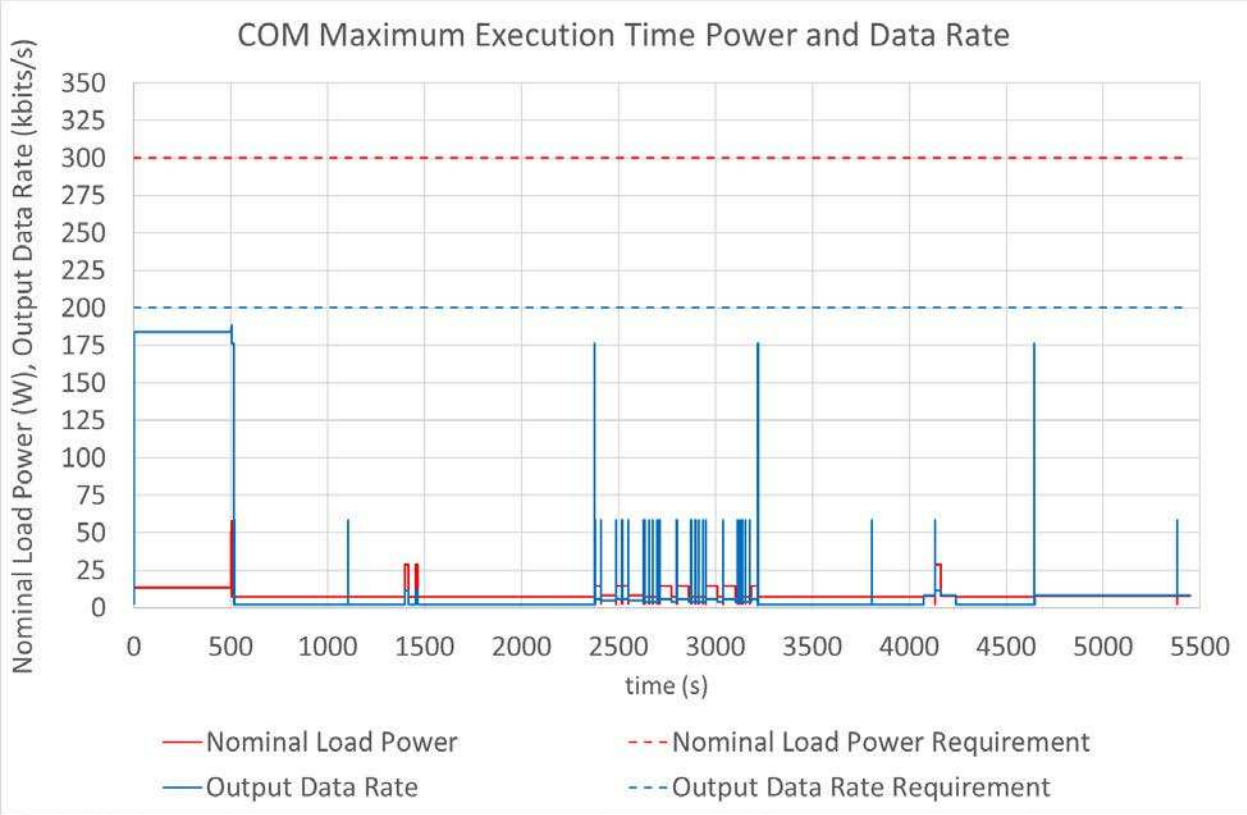


FIGURE 53. NON-MBSE COM MAXIMUM EXECUTION TIME POWER AND DATA RATE RESULTS.

In the MBSE approach, the system simulation automatically generated the virtual system through instantiating element blocks, reading and initiating attribute values, and executing activities. Results from the simulation were also automatically generated and displayed on Time Series Charts. Figure 54, Figure 55, and Figure 56 show the MBSE approach time series charts displaying the results from the minimum, average, and maximum execution times.

FIGURE 54. MBSE COM MINIMUM EXECUTION TIME POWER AND DATA RATE RESULTS.

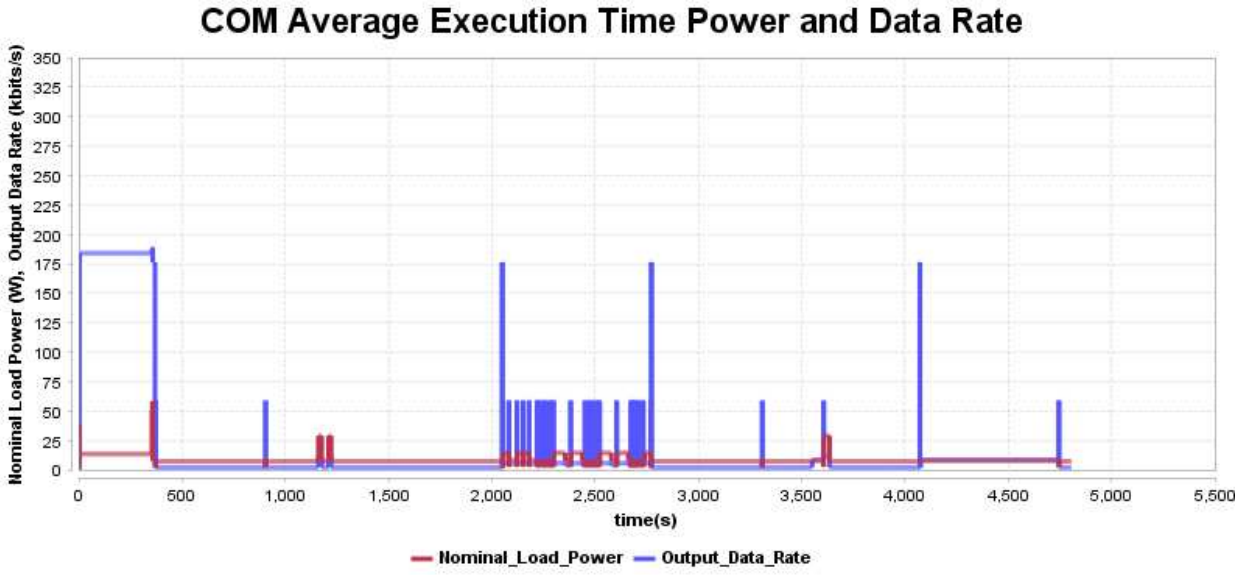


FIGURE 55. MBSE COM AVERAGE EXECUTION TIME POWER AND DATA RATE RESULTS.

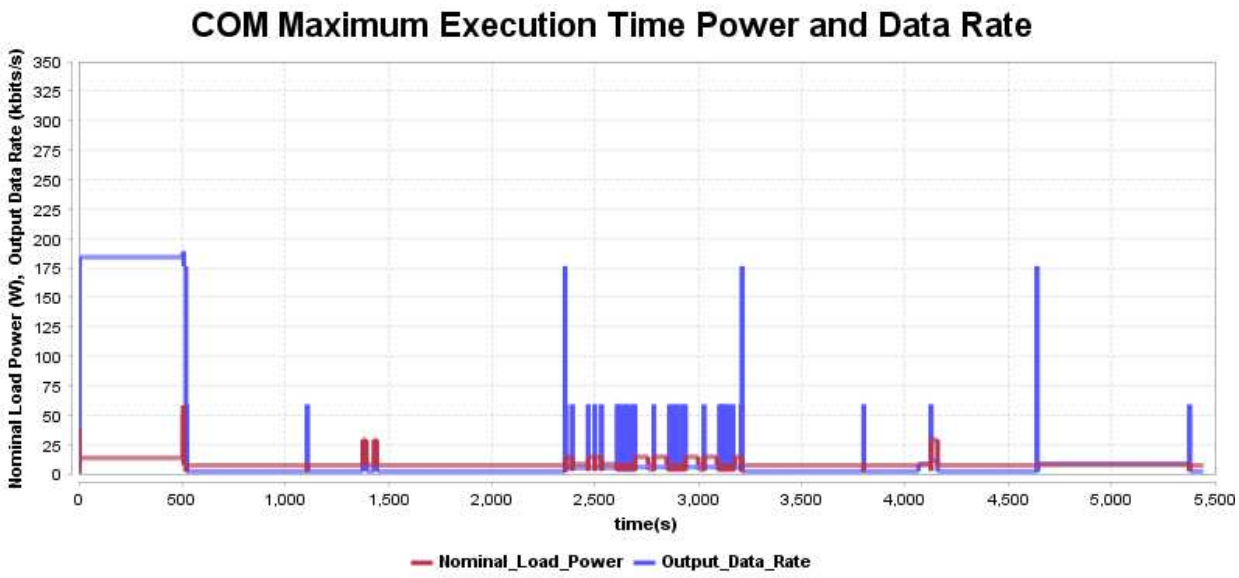


FIGURE 56. MBSE COM MAXIMUM EXECUTION TIME POWER AND DATA RATE RESULTS.

Integrating the virtual system elements was carried out manually in the Non-MBSE approach through generating a copy of the computerized model spreadsheet. This included copying over the structural, behavioral, and parametric relationships captured in the spreadsheet.

In the MBSE approach, virtual system integration was performed automatically. When the top-level system was simulated, lower-level bocks, along with their attributes and relationships were automatically instantiated. When top-level activities were executed, lower-level activities were automatically linked to these activities and called during execution. Additionally, parameter relationships were automatically generated and rolled up through the pre-defined rollup patterns during simulation.

The computerized model was manually validated in both the Non-MBSE and MBSE approaches. Computerized model validation was performed using various techniques, including walk throughs of the computerized model, operational graphics (looking at power and output data rate profiles in the Time Series Charts to ensure they behaved correctly), and parameter

variability-sensitivity analysis (varying control and noise variables to determine if the directionality of the change in the output were logical).

Requirements verification was performed using the simulation output results. In the Non-MBSE approach, this was carried out manually by visually evaluating the power and data output rate profiles in the Time Series Charts and comparing the values to the not-to-exceed limits in the requirements. From evaluating the Non-MBSE approach time series charts in Figure 51, Figure 52, and Figure 53, all values in the plots remain visibly below the dashed lines representing the Nominal Load Power and Output Data Rate requirement limits, indicating the that COM meets these two requirements.

In the MBSE approach, requirements verification was carried out automatically during simulation by Cameo Systems Modeler. From evaluating the MBSE approach time series charts in Figure 54, Figure 55, and Figure 56, there are no requirement violation alerts displayed by Cameo on the chart, indicating the that COM meets these two requirements.

To illustrate how requirements violations are indicated in the Non-MBSE and MBSE time series charts, two additional charts were generated with an artificially increased Outward Vision System Output Data Rate attribute value. Figure 57 shows a Non-MBSE example of the COM Output Data Rate exceeding requirement with an artificially increased Outward Vision System Output Data Rate, as indicated by the green Output Data Rate values visibly above the green dashed line representing the Output Data Rate requirements limit. Figure 58 shows an MBSE example of the COM Output Data Rate exceeding requirement with an artificially increased Outward Vision System Output Data Rate, as indicated by the red shaded regions automatically displayed on the chart.

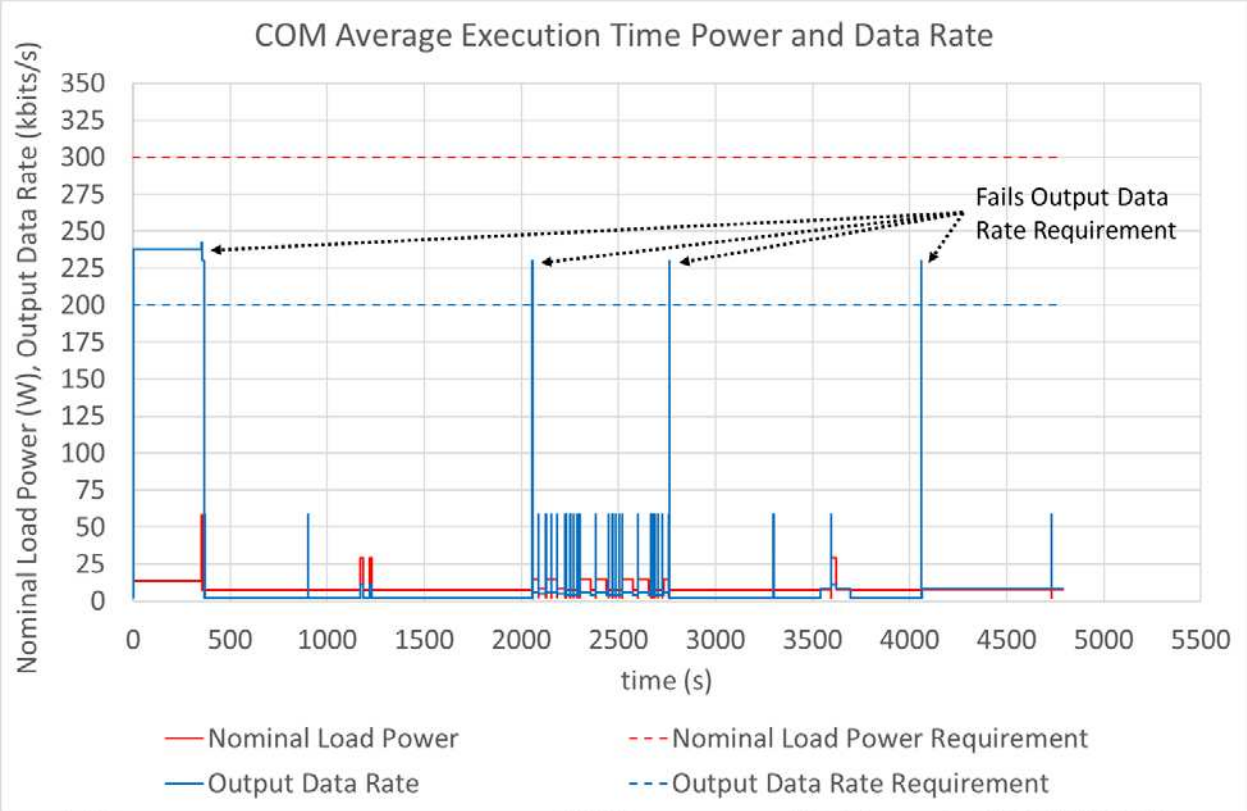


FIGURE 57. NON-MBSE EXAMPLE OF COM OUTPUT DATA RATE FAILING REQUIREMENT WITH AN ARTIFICIALLY INCREASED OUTWARD VISION SYSTEM OUTPUT DATA RATE TO DEMONSTRATE VISUAL REQUIREMENTS VERIFICATION USING THE NON-MBSE DOCUMENT.

FIGURE 58. MBSE EXAMPLE OF COM OUTPUT DATA RATE FAILING REQUIREMENT WITH AN ARTIFICIALLY INCREASED OUTWARD VISION SYSTEM OUTPUT DATA RATE TO DEMONSTRATE AUTOMATIC REQUIREMENTS VERIFICATION USING THE MBSE TOOL. THE SHADED REGIONS OF THE PLOT LABELED "REQ 2 FAILS" INDICATES WHEN THE COM OUTPUT DATA RATE REQUIREMENT (REQUIREMENT ID "2") WAS NOT MET DURING THE SIMULATION.

Since the physical system itself had not yet been produced at this early Pre-Phase A stage of the project, the virtual system was not qualified in this research. However, this activity would take place at a later phase when a physical system is produced and available for testing.

6.3.6. Quantification of the Number of Knowledge Elements Manually and Automatically Processed

To quantify the work carried out during each activity within the modeling and simulation process, the key elements of knowledge processed in each of the activities were identified and counted. Table 22 lists out the key types of knowledge relevant for the modeling and simulation needed to verify the COM power and output data rate requirements. Table 23 maps out which modeling and simulation activities each of the knowledge elements were used in. Table 24 provides a count of each know element at each of the system levels. A total of 1,090 key knowledge elements were identified by the team and tracked through the modeling and

simulation activities. These values were determined by manually counting unique model elements and attributes common to the Non-MBSE and MBSE system model diagrams, parameter diagram, experiment matrix, simulation configurations, and time series charts.

TABLE 22. MODELING AND SIMULATION KEY KNOWLEDGE ELEMENTS AND DEFINITIONS.

Knowledge Category	Knowledge	Definition
System Structure	Element Type	Structural element of the system
	Structural Decomposition	Whole-part relationship between elements of higher and lower hierarchy levels
	Attribute Value	Value assigned to a parameter of a system element
	Mathematical Rule	Mathematical relationship between parameters, including roll-up patterns
System Behavior	Functional Decomposition	Function-subfunction relationship indicating a subfunction is necessary to accomplish a function
	Function	Action a system element performs to fulfill its requirements
	Function Duration Value	Value assigned to the duration of function
	Control Flow	Relationship between actions describing flow of control from one action to the next
	Functional Allocation	Relationship that assigns a function to a system element
System Requirements	Requirement	Agreed-upon need, desire, want, capability, capacity, demand, or constraint of a system element
	Requirement Text	Shall statement that includes the performing element, a "shall", and a function, measurable performance property, or constraint
	Requirement Parent	Parent-child relationship between requirements
	Requirement Allocation	Relationship that assigns a requirement to a system element
Experiments	Experiment Scenario	Sequence of system functions for a particular use case under specific conditions
	Simulation Configuration Attribute Value	Attribute that specifies the model, data, method, implementation, and realization of the simulation
	Time Series Chart Attribute Value	Attribute that specifies the output and reporting format
Experiment Parameters	Input	Signal that activates a system function during a virtual system experiment
	Noise Factor	System uncertainty for a virtual system experiment
	Control Factor	System specification for a virtual system experiment
	Output	Output data set for a virtual system experiment

TABLE 23. MODELING AND SIMULATION KNOWLEDGE TO ACTIVITY MAPPING.

		Modeling and Simulation Activity												
		Analyze	Model	Decompose Model	Validate Conceptual Model	Program	Integrate Model	Verify Computerized Model	Validate Computerized Model	Plan Experiments	Simulate	Integrate Virtual System	Qualify Virtual System	Virtually Verify Requirements
Modeling and Simulation Knowledge	Element Type	●	●		●	●		●			●			
	Structural Decomposition	●		●	●		●	●				●		
	Attribute Value	●	●		●	●		●			●			
	Mathematical Rule	●		●	●		●	●				●		
	Functional Decomposition	●		●	●		●	●				●		
	Function	●	●		●	●		●			●			
	Function Duration Value	●	●		●	●		●			●			
	Control Flow	●	●		●	●		●			●			
	Functional Allocation	●	●		●	●		●			●			
	Requirement	●	●		●	●		●			●			●
	Requirement Text	●	●		●	●		●			●			●
	Requirement Parent	●		●	●		●	●				●		
	Requirement Allocation	●	●		●	●		●			●			
	Experiment Scenario									●	●			
	Simulation Configuration Attribute Value									●	●			
	Time Series Chart Attribute Value									●	●			
	Input									●	●			
	Noise Factor									●	●			
	Control Factor									●	●			
	Output								●	●	●		●	●

TABLE 24. MODELING AND SIMULATION KNOWLEDGE.

		Element Type	Structural Decomposition	Attribute Value	Mathematical Rule	Functional Decomposition	Function	Function Duration Value	Control Flow	Functional Allocation	Requirement	Requirement Text	Requirement Parent	Requirement Allocation	Experiment Scenario	Simulation Configuration Attribute Value	Time Series Chart Attribute Value	Input	Noise Factor	Control Factor	Output	**Total**
Organization	Module Level (L4)	1	0	2	0	0	10	30	10	10	2	2	0	2	3	12	48	3	3	6	6	**150**
	Subystem Level (L5)	4	4	8	16	16	16	48	16	16	0	0	0	0	0	0	0	0	0	0	0	**144**
	Assembly Level (L6)	15	15	30	92	92	92	276	92	92	0	0	0	0	0	0	0	0	0	0	0	**796**
	Total	**20**	**19**	**40**	**108**	**108**	**118**	**354**	**118**	**118**	**2**	**2**	**0**	**2**	**3**	**12**	**48**	**3**	**3**	**6**	**6**	**1090**

Based on the quantities in Table 24, along with the mapping of knowledge types to activities in Table 23, the numbers of knowledge elements processed during each modeling and simulation activity at each system level were calculated, as presented in Table 25 and Table 26. Where activities were not performed at a specific system level, an "N/A" was entered. For example, since Level 4 (the COM system level) was the highest level modeled and simulated, no relationships to higher level elements were needed to be modeled or simulated. Additionally, only two Level 4 requirements we verified in this modeling and simulation effort (the COM power and data rate). Therefore, no experiments were planned nor requirements verified at the lower Level 5 or 6 system levels.

TABLE 25. KNOWLEDGE PROCESSING FOR THE NON-MBSE APPROACH. RED-COLORED CELLS INDICATE WHERE KNOWLEDGE ELEMENTS WERE MANUALLY PROCESSED.

		Modeling and Simulation Activity												
		Analyze	Model	Decompose Model	Validate Conceptual Model	Program	Integrate Model	Verify Computerized Model	Validate Computerized Model	Plan Experiments	Simulate	Integrate Virtual System	Qualify Virtual System	Virtually Verify Requirements	**Total**
Organization	Module Level (L4)	69	69	N/A	69	69	N/A	69	6	81	150	N/A	N/A	10	**592**
	Subystem Level (L5)	130	108	22	130	108	22	130	24	N/A	108	22	N/A	N/A	**804**
	Assembly Level (L6)	712	597	115	712	597	115	712	90	N/A	597	115	N/A	N/A	**4362**
	Total	**911**	**774**	**137**	**911**	**774**	**137**	**911**	**120**	**81**	**855**	**137**	**0**	**10**	**5758**

TABLE 26. KNOWLEDGE PROCESSING FOR THE MBSE APPROACH. RED-COLORED CELLS INDICATE WHERE KNOWLEDGE ELEMENTS WERE MANUALLY PROCESSED. GREEN-COLORED CELLS INDICATE WHERE KNOWLEDGE ELEMENTS WERE AUTOMATICALLY PROCESSED.

		Modeling and Simulation Activity												
		Analyze	Model	Decompose Model	Validate Conceptual Model	Program	Integrate Model	Verify Computerized Model	Validate Computerized Model	Plan Experiments	Simulate	Integrate Virtual System	Qualify Virtual System	Virtually Verify Requirements	**Total**
Organization	Module Level (L4)	69	69	N/A	69	69	N/A	69	6	81	150	N/A	N/A	10	**592**
	Subystem Level (L5)	130	108	22	130	108	22	130	24	N/A	108	22	N/A	N/A	**804**
	Assembly Level (L6)	712	597	115	712	597	115	712	90	N/A	597	115	N/A	N/A	**4362**
	Total	**911**	**774**	**137**	**911**	**774**	**137**	**911**	**120**	**81**	**855**	**137**	**0**	**10**	**5758**

Modeling and simulation activities that needed to be carried out manually were identified and are highlighted in red in Table 25 and Table 26 for the Non-MBSE and MBSE approaches. For the Non-MBSE approach, all activities were completed manually on individual documents. For the MBSE approach, several of the modeling and simulation activities were automatically completed, as highlighted in green in Table 26. This is due to the executability of the SysML

conceptual model, which can automatically translate the conceptual model into both the computerized model and virtual system. Activities that required analysis of the system, initial modeling of the conceptual model, evaluation to validate the models, and planning of the experiments, still needed to be performed manually. The total number of manual and automatic knowledge processing from Table 25 and Table 26 were calculated and summarized in Table 27.

TABLE 27. COMPARISON OF AUTOMATION OF KNOWLEDGE PROCESSING FOR MBSE AND NON-MBSE APPROACHES.

		Automation		
		Automated	Manual	**Total**
Approach	Non-MBSE	0	5758	**5758**
	MBSE	2824	2934	**5758**

Figure 59 and Figure 60 summarize the modeling and simulation activities and relative quantities of manual and automatic knowledge processing for the Non-MBSE and MBSE approaches used to simulate COM power and data rate. Since only Level 4 requirements (COM power and data rate) were simulated and verified, the Plan Experiments and Virtually Verify Requirements activities associated with the Virtual System Experiments were only carried out at Level 4. Additionally, since the Physical System was not yet developed nor Physical System tests performed, no Qualify Virtual System activities were carried out with the Physical System Tests. Activities colored in red were manually carried out in the approach, while activities colored in green were automatically carried out in the approach. The thickness of the activity lines provides a scaled measure of how much total knowledge was processed during each activity, ranging from 6 knowledge elements (thinnest lines) to 712 knowledge elements (thickest lines).

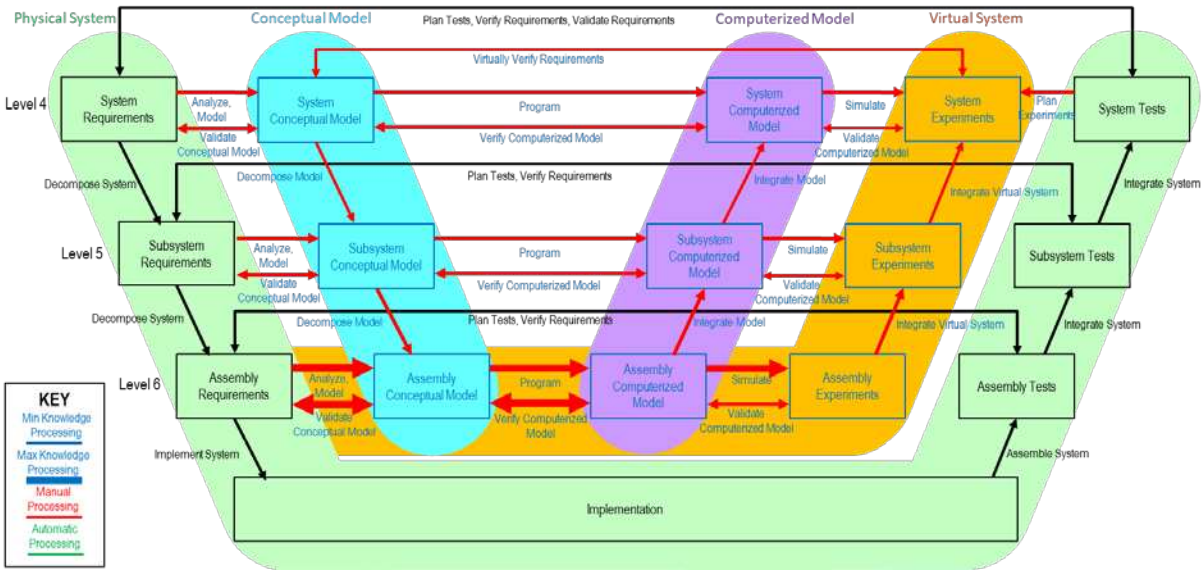


FIGURE 59. RELATIVE QUANTITIES OF MANUAL AND AUTOMATIC KNOWLEDGE PROCESSING ASSOCIATED WITH THE COM MODELING AND SIMULATION ACTIVITIES USING THE NON-MBSE APPROACH.

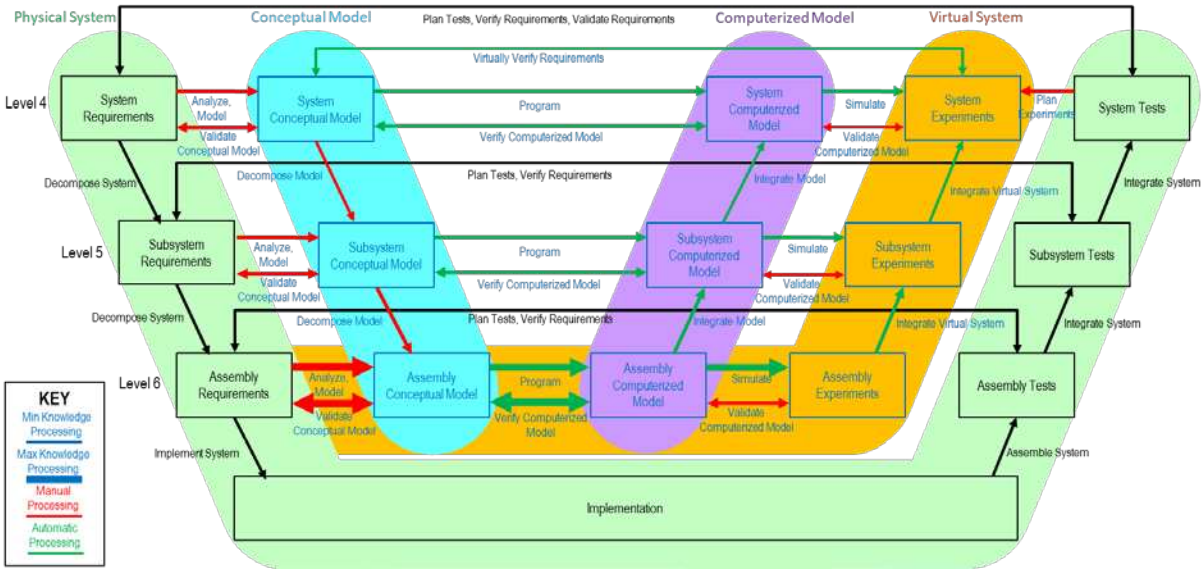


FIGURE 60. RELATIVE QUANTITIES OF MANUAL AND AUTOMATIC KNOWLEDGE PROCESSING ASSOCIATED WITH THE COM MODELING AND SIMULATION ACTIVITIES USING THE MBSE APPROACH.

6.4. Results and Discussion

MBSE is claimed to offer user experience benefits through its ability to 1) provide a higher level of support for automation, 2) reduce the burden of systems engineering tasks, and 3) reduce effort [129]. This section discusses and quantifies how the MBSE approach explored in this research provides user experience benefits within these three categories within the context of modeling and simulating the COM robotic space system architecture. Following this discussion, recommendations for future work are also presented.

6.4.1. Higher Level of Support for Automation

Automation is defined as the allocation of functions to machines versus humans [77]. Automated simulation with MBSE enables engineering teams to reduce development cycle time through integrating workflow processes in an automated environment, as well as deliver more reliable, better-quality products through accelerated evaluation of design alternatives [52].

The amount of manual and automatic knowledge processing for each of the modeling and simulation activities in the Non-MBSE and MBSE approaches is shown in Table 25 and Table 26. The total number of manual and automatic knowledge processing is summarized in Table 27. In the Non-MBSE approach, all knowledge processing was performed manually. In the MBSE approach, 2,824 of 5,758 total knowledge processing was automated (49% of the knowledge processing). Therefore, a higher level of support for automation is apparent in the MBSE approach relative to the Non-MBSE approach.

6.4.2. Reduced Burden of Systems Engineering Tasks

Systems engineering tasks have been shown to have a quantifiable positive correlation to program success [153]. However, it is rare for a program to achieve all systems engineering tasks and contributions before resource limits are met [31]. Modeling and simulation tasks are specific

systems engineering tasks applicable to key systems engineering processes such as product verification and validation [77]. Progress has been made towards integrating modeling and simulation with MBSE languages, which allows simulation semantics directly expressed within the model to be executed via model transformation to a machine-readable language [140]. This progress can be leveraged to carry out modeling and simulation tasks that would otherwise need to be performed manually, reducing this burden from the engineering team.

The modeling and simulation activities were all performed manually in the Non-MBSE approach. The MBSE approach automated a number of the modeling and simulation activities, which included the Program, Integrate Model, Verify Computerized Model, Simulate, Integrate Virtual System, and Virtually Verify Requirements activities. As shown in Table 26, a total of 14 instances of these activities carried out over the three system levels were automated, representing a reduced burden of 14 systems engineering task instances for the engineering team with the MBSE approach relative to the Non-MBSE approach.

6.4.3. Reduced Effort

Mental effort is a neurocognitive process that reflects the controlled expenditure of psychological information-processing resources during perception, cognition, and action [154]. Cognitive load, more specifically, is the mental effort made by a person to perform a task [155]. Intrinsic cognitive load is the cognitive load that is inherent to the task, defined by the intrinsic task complexity [155]. Cognitive overload can have detrimental effects, including impaired performance and decision-making, stress, difficulty in retrieving knowledge, creativity impedance, and difficulty with analyzing and organizing knowledge [156].

To assess the intrinsic complexity of a task to assess cognitive load, tasks can be classified by the specific cognitive processes they utilize, and then assessed for complexity based on the

complexity of the cognitive process. The cognitive process dimension of the Revised Bloom's Taxonomy [102] was chosen as a means for classifying the cognitive process required for each of the modeling and simulation activities. The Revised Bloom's Taxonomy has its basis in modern cognitive science and cognitive psychology. The cognitive process dimension of the taxonomy provides a comprehensive set of 19 cognitive processes grouped into six process categories. The cognitive processes are organized on a continuum of increasing complexity, with the "remember" category being least complex, and "create" category being most complex. Table 28 provides an overview of the Revised Bloom's Taxonomy cognitive process dimension.

TABLE 28. REVISED BLOOM'S TAXONOMY COGNITIVE PROCESSES AND DEFINITIONS [102].

Greater Level of Complexity

Category	Cognitive Process	Definition
Remember	Recognizing	Locating knowledge in memory similar to presented material
	Recalling	Retrieving knowledge from memory
Understand	Interpreting	Converting knowledge from one form to another
	Exemplifying	Providing specific examples of a concept or principle
	Classifying	Determining something belongs to a particular category based on a general concept or principle
	Summarizing	Constructing a representative statement of given information or abstracting a general theme
	Inferring	Finding a pattern amongst a set of instances or drawing a conclusion
	Comparing	Detecting similarities or differences between objects or ideas
	Explaining	Constructing a cause-and-effect model of a system
Apply	Executing	Applying a procedure to a familiar task
	Implementing	Applying a procedure to an unfamiliar task
Analyze	Differentiating	Distinguishing relevant from irrelevant parts of presented material
	Organizing	Determining how elements fit or function within a structure
	Attributing	Determine an underlying point of view, bias, values, or intent of presented material
Evaluate	Checking	Determining inconsistencies of fallacies within a process or product
	Critiquing	Detecting inconsistencies between a product and external criteria
Create	Generating	Coming up with alternative hypotheses based on criteria
	Planning	Devising a procedure for accomplishing some task
	Producing	Inventing a product

Table 29 shows a mapping of the modeling and simulation activities to the Revised Bloom's Taxonomy cognitive processes. The activities were categorized based on the cognitive process

definitions defined in the Revised Bloom's Taxonomy book by Anderson et al. [102]. As shown in Table 29, a majority of the modeling and simulation activities fall into the Analyze and Evaluate cognitive process categories, which represent higher complexity on the cognitive process complexity continuum.

TABLE 29. REVISED BLOOM'S TAXONOMY COGNITIVE PROCESS TO ACTIVITY MAPPING.

			Modeling and Simulation Activity												
			Analyze	Model	Decompose Model	Validate Conceptual Model	Program	Integrate Model	Verify Computerized Model	Validate Computerized Model	Plan Experiments	Simulate	Integrate Virtual System	Qualify Virtual System	Virtually Verify Requirements
Revised Bloom's Taxonomy Cognitive Process	**Remember**	Recognizing													
		Recalling													
	Understand	Interpreting		●			●								
		Exemplifying													
		Classifying													
		Summarizing													
		Inferring													
		Comparing													
		Explaining													
	Apply	Executing										●			
		Implementing													
	Analyze	Differentiating	●												
		Organizing						●					●		
		Attributing			●										
	Evaluate	Checking							●						●
		Critiquing				●				●				●	
	Create	Generating													
		Planning									●				
		Producing													

Table 30 and Table 31 show the quantity of knowledge elements processed manually and automatically for both the Non-MBSE and MBSE approaches based on the total quantities for each activity as reported in Table 25 and Table 26, along with the activity to cognitive process

mapping in Table 29. Based on the Total Automated quantities in Table 26, the MBSE approach reported automation of activities in the Understand, Apply, Analyze, and Evaluate cognitive process categories. The cognitive process with the highest number of automated knowledge processing fell within the Checking cognitive process, which was associated with model verification. Overall, the MBSE approach reduced the amount of knowledge processing in six modeling and simulation activities spanning four cognitive processes, ranging from the lower to higher range of complexity. This in turn, point to a reduction in cognitive load amongst these six modeling and simulation activities, leading to a reduction in overall mental effort for the engineering team when applying the MBSE approach, with the greatest relative reduction in mental effort associated with model verification activities.

TABLE 30. MANUAL AND AUTOMATED KNOWLEDGE PROCESSING ORGANIZED BY COGNITIVE PROCESS FOR THE NON-MBSE APPROACH. RED-COLORED CELLS INDICATE WHERE KNOWLEDGE ELEMENTS WERE MANUALLY PROCESSED. GREEN-COLORED CELLS INDICATE WHERE KNOWLEDGE ELEMENTS WERE AUTOMATICALLY PROCESSED.

			Modeling and Simulation Activity														
			Analyze	Model	Decompose Model	Validate Conceptual Model	Program	Integrate Model	Verify Computerized Model	Validate Computerized Model	Plan Experiments	Simulate	Integrate Virtual System	Qualify Virtual System	Virtually Verify Requirements	**Total Manual**	**Total Automated**
Cognitive Process	**Understand**	Interpreting		774			774									**1548**	**0**
	Apply	Executing										855				**855**	**0**
	Analyze	Differentiating	911													**911**	**0**
		Organizing						137					137			**274**	**0**
		Attributing			137											**137**	**0**
	Evaluate	Checking							911						10	**921**	**0**
		Critiquing				911				120				0		**1031**	**0**
	Create	Planning									81					**81**	**0**
		Total	**911**	**774**	**137**	**911**	**774**	**137**	**911**	**120**	**81**	**855**	**137**	**0**	**10**	**5758**	**0**

TABLE 31. MANUAL AND AUTOMATED KNOWLEDGE PROCESSING ORGANIZED BY COGNITIVE PROCESS FOR THE MBSE APPROACH. RED-COLORED CELLS INDICATE WHERE KNOWLEDGE ELEMENTS WERE MANUALLY PROCESSED. GREEN-COLORED CELLS INDICATE WHERE KNOWLEDGE ELEMENTS WERE AUTOMATICALLY PROCESSED.

			Modeling and Simulation Activity														
			Analyze	Model	Decompose Model	Validate Conceptual Model	Program	Integrate Model	Verify Computerized Model	Validate Computerized Model	Plan Experiments	Simulate	Integrate Virtual System	Qualify Virtual System	Virtually Verify Requirements	**Total Manual**	**Total Automated**
Cognitive Process	**Understand**	Interpreting		774			774									**774**	**774**
	Apply	Executing										855				**0**	**855**
	Analyze	Differentiating	911													**911**	**0**
		Organizing						137					137			**0**	**274**
		Attributing			137											**137**	**0**
	Evaluate	Checking							911						10	**0**	**921**
		Critiquing				911				120				0		**1031**	**0**
	Create	Planning									81					**81**	**0**
		Total	**911**	**774**	**137**	**911**	**774**	**137**	**911**	**120**	**81**	**855**	**137**	**0**	**10**	**2934**	**2824**

6.4.4. Future Work

The modeling and simulation process performed in this research was limited to a specific subsystem (the COM), addressed only two requirements (nominal load power and output data rate), and solely used SysML and Cameo Systems Modeler to implement modeling and simulation activities in the MBSE approach. For future work, recommendations are test the process on additional systems, on a greater variety of requirements, and with different MBSE languages and tools:

- Testing the process on additional systems: Applying the modeling and simulation process using both the MBSE and non-MBSE approaches on additional systems can help validate the process across additional engineering domains, as well as gather additional data on the benefits provided by MBSE in modeling and simulation.

- Testing the process on a greater variety of requirements: Parameter-based requirements that are relatively straightforward to represent in models and make use of available functions build into the simulation tool (e.g., built-in rollup patterns in Cameo Systems Modeler) can be easily modeled, simulated, and verified through analysis in a virtual environment. Other requirements may not be easily modeled, simulated, and verified in a virtual environment, or may only be verifiable through inspection, demonstration or test. Testing out the modeling and simulation process on a wider set of requirements will help both assess the extent of the MBSE approach to verify system requirements, as well as understand its limitations in the types of requirements that can be modeled, simulated, and verified.
- Testing the process with different MBSE languages and tools: Different languages possess unique constructs to incorporate simulation semantics into the model. Different tools possess unique build-in patterns, abilities to transform model syntax into a solver or other machine-readable language, execute the simulation, and perform subsequent analysis on the results. Through testing the process with different MBSE languages and tools, additional benefits of the MBSE approach may be identified.

6.5. Conclusion

MBSE case studies in the literature assert that there are benefits to MBSE when applied to modeling and simulation of space systems. However, there is a lack of side-by-side comparisons that provide quantitative evidence of the advantages MBSE approaches have over traditional, document-based approaches. The COM Pre-Phase A architectural was used as a case study to evaluate the benefits of an MBSE modeling and simulation approach over a traditional, non-MBSE approach. The COM architecture was modeled, simulated, and evaluated against nominal

load power and output data rate requirements to compare the two approaches. User experience benefits with system modeling and simulation were explored and measured in terms of automated knowledge processing.

A new modeling and simulation-centric V-model was synthesized for this case study from existing V-model and modeling and simulation process models to map out the modeling and simulation activities within the context of the system development lifecycle and used as a tool to compare the Non-MBSE and MBSE approaches. The V-model contains four system representations consisting of the physical system, the conceptual model, the computerized model, and the virtual system. A set of activities connect the elements of the model to carry out modeling, simulation, and system development.

A three-phase modeling and simulation process was used to model and simulate the COM. This process consisted of an analysis and modeling phase, computer programming and implementation phase, and experimentation phase. The process was followed using a Non-MBSE and an MBSE approach. The total number of manual and automatic knowledge processed was calculated for both approaches and used to quantify the benefits of the MBSE approach relative to the Non-MBSE approach. A total of 1,090 key knowledge elements were identified and tracked through the modelling and simulation process. The total number of manual and automatic knowledge processing was calculated for both approaches. In the Non-MBSE approach, all 5,758 knowledge elements were manually processed during the modeling and simulation process. In the MBSE approach, 2,824 of 5,758 total knowledge processing was automated (49% of the knowledge processing).

The MBSE approach showed user experience benefits with modeling and simulation of the COM robotic space system through providing a higher level of support for automation, reducing

the burden of systems engineering tasks, and reducing effort. Recommendations for future work include testing the modeling and simulation process on additional systems, on a greater variety of requirements, and with different MBSE languages and tools.

7. CONCLUSIONS

The purpose of this research was to investigate and document the benefits of MBSE in architecting robotic space systems, particularly with describing, developing, and evaluating the system architecture. These areas were addressed through three research questions, which were individually explored in Chapters 4, 5, and 6, using the COM Pre-Phase A architecture development as a case study:

1) **Can an MBSE approach better capture the information content required to describe a robotic space system architecture relative to a non-MBSE approach, as assessed by a higher categorized fraction of architecture content that can be formally captured in the appropriate knowledge category with the language and tool?**
2) **Can an MBSE approach reduce the implementation effort required to develop a robotic space system architecture relative to a non-MBSE approach, as assessed by a higher quantity of information transfer between tasks that can be automated for carrying out the architecting process?**
3) **Can an MBSE approach more efficiently evaluate a robotic space system architecture than a non-MBSE approach, as assessed by a higher quantity of knowledge that can be automatically processed during modeling and simulation activities for system requirements verification?**

Chapter 4 explored Research Question 1, assessing how MBSE increased information capture of the COM architecture descriptive model. The types and quantities of knowledge that were captured by an MBSE approach and a traditional architecting approach were compared to

measure the benefits of the MBSE approach in managing the complexity of a robotic space system. An architecture framework was established, covering system, subsystem, and assembly levels, along with structure, behavior, data, and requirements perspectives. A total of 4,389 knowledge elements were classified using the Revised Bloom's Taxonomy knowledge dimension and used to quantitatively compare the two approaches. Out of the 4,389 knowledge elements that made up the COM architecture, the MBSE approach was able to represent 3,879 of them, while the non-MBSE approach was able to represent 3,076 of them, providing an 18% increase in knowledge representation relative to the non-MBSE approach. The MBSE approach more completely captured the COM architectural knowledge than the non-MBSE approach by providing a higher fraction of architecture content in the appropriate knowledge category.

Chapter 5 explored Research Question 2, assessing how MBSE reduced implementation effort during the COM architecting process. The quantity of information that was automatically transferred through the associations generated using an MBSE approach versus a traditional systems engineering approach was investigated to measure the benefits of MBSE in architecting a robotic space system. The approaches were analyzed using design structure matrices (DSM) and evaluated based on the amount of information transferred between process tasks manually (e.g., elements physically typed into text boxes in a presentation slide) vs. automatically (e.g., elements automatically filled out within a block in a model view due to explicitly defined element associations). A total of 4,819 information element transfers were traced in DSMs and used to quantitatively compare the two approaches. The non-MBSE approach required manual transfer for all 4,819 information elements. The MBSE approach required manual transfer for 4,189 information elements and automatic transfer for 630 information elements, providing a 13% increase in the automation of information transfer relative to the non-MBSE approach.

Additionally, an extensive MBSE approach that further utilizes MBSE resources for the trade study and peer review tasks was predicted to further increase automation to 3,927 of 5,758 knowledge elements (81% of knowledge processing). The MBSE approach provided a slight reduction in the implementation effort required to develop the COM architecture relative to a non-MBSE approach by providing a higher quantity of automatic information transfer between architecting tasks.

Chapter 6 explored Research Question 3, assessing how MBSE improved evaluation efficiency when modeling and simulating the COM architecture. The benefits of an MBSE modeling and simulation approach over a traditional, non-MBSE approach were investigated. The COM architecture was modeled, simulated, and evaluated against Nominal Load Power and Output Data Rate requirements to compare the two approaches. A new modeling and simulation-centric V-model was synthesized from existing V-model and modeling and simulation process models to map out the modeling and simulation activities within the context of the system development lifecycle and used as a tool to compare the non-MBSE and MBSE approaches. A three-phase modeling and simulation process consisting of an analysis and modeling phase, computer programming and implementation phase, and experimentation phase was used to model and simulate the COM. The total number of manual and automatic knowledge elements processed was calculated for both approaches and used to quantify the benefits of the MBSE approach relative to the Non-MBSE approach. In the non-MBSE approach, all 5,758 knowledge elements were manually processed during the modeling and simulation process. In the MBSE approach, 2,824 of 5,758 total knowledge processing was automated (49% of the knowledge processing). The MBSE approaches improved evaluation efficiency of the COM architecture

relative to the non-MBSE approach by providing a higher quantity of automatic knowledge processing during modeling and simulation activities.

The combination of results from the investigations carried out though the three research questions addresses the overarching research question:

What measurable advantages does MBSE provide to architecting robotic space systems from a knowledge capture and process implementation viewpoint?

The MBSE approach provided measurable advantages to architecting the COM robotic space system in terms of a higher fraction of formally captured architecture content in the appropriate knowledge category, a higher quantity of automatic information transfer between architecting process tasks, and a higher quantity of automatic knowledge processing during modeling and simulation process activities. The overall results demonstrate the benefits of MBSE in managing the complexity of robotic space systems and strengthen the case for adopting MBSE within the systems engineering community.

Despite the advantages illustrated above, several disadvantages were observed. Due to the larger quantity of knowledge explicitly represented in the MBSE approach, a greater amount of time, effort, and resources were required to set up the model and architect the system than the non-MBSE approach. Additionally, limitations in the value added by the MBSE approach were observed in terms of:

- Knowledge representation in the architecture descriptive model: Only an 18% improvement in knowledge representation was observed using MBSE. 12% of the total knowledge was still not fully represented in the MBSE architecture descriptive model.
- Automated information transfer in the architecting process: Only a 13% improvement in automated information transfer was observed using MBSE. 87% of information transfer

still required manual transfer in the MBSE architecting process. With the extensive MBSE approach, an 81% improvement in automated information transfer was predicted. 19% of information transfer still required manual transfer in the extensive MBSE architecting process.

- Automated knowledge processing in the evaluation of the architecture: Only a 49% improvement in knowledge processing was observed using MBSE. 51% of knowledge processing still required manual processing in the MBSE modeling and simulation process.

7.1. Research Contributions

Model-based Systems Engineering (MBSE) case studies in the literature assert that there are benefits to MBSE when architecting complex systems. However, there is an absence of case studies in the literature providing side-by-side comparisons of projects using both MBSE and traditional, non-MBSE approaches with quantitative evidence that measures the relative advantage of one approach over the other. In particular, there is a lack of quantitative comparative analysis studies of MBSE applied to robotic space systems during the architecture phase of the project, where MBSE promises to provide the most value.

This dissertation addressed these gaps in the literature and makes the following research contributions:

- A case study providing side-by-side comparisons of MBSE and non-MBSE approaches to describe, develop, and evaluate a robotic space system with quantitative evidence of the relative advantages of an MBSE approach over a non-MBSE approach.
- An architecture framework for describing a robotic space system that defines the structure, data, behavior, and requirements at the system, subsystem and assembly levels.

- An analysis of MBSE from a cognitive psychological knowledge perspective, providing quantitative evidence that MBSE tools capture higher-level, more abstract knowledge than traditional, document-based systems engineering tools.
- Quantitative evidence of the benefits of an MBSE approach over a non-MBSE approach for describing a robotic space system in terms of categorized fractions of architecture content.
- An MBSE approach for architecting a robotic space system that defines the structure, data, behavior, and requirements of the system at each organization level and incorporates trade studies and peer reviews between levels.
- A methodology that uses DSMs as a tool to quantitatively measure the benefits of an MBSE approach relative to a non-MBSE approach.
- Quantitative evidence of the benefits of an MBSE approach over a non-MBSE approach for architecting a robotic space system in terms of automatic information transfer.
- A modeling and simulation-centric interpretation of the canonical systems engineering V-model that incorporates modeling and simulation elements and activities into the system development lifecycle, and enables comparison of modeling and simulation approaches.
- Quantitative evidence of the benefits of an MBSE approach over a non-MBSE approach for architecture modeling and simulation in terms of automatic knowledge processing.

7.2. Future Work

The scope of this research was limited to fit within the available project resource constraints and operate within the given project environment. This case study was limited to the COM robotic space system architecture during Pre-Phase A and carried out by a single development team at JPL, which applied the MBSE and non-MBSE approaches in parallel. The MBSE

approach used SysML and Cameo Systems Modeler as its sole MBSE language and tool. In describing the architecture, the MBSE descriptive model did not capture all high-level knowledge, such as design principles, design approaches and rationales, risk, development strategies and rationales, organizational core competencies, and requirement verification methods. In developing the architecture, the MBSE architecting process approach only utilized MBSE resources during the architecture definition tasks, while still utilizing non-MBSE resources (documents) for trade study and peer review tasks. In evaluating the architecture, the MBSE simulation only verified a limited number of requirements (nominal load power and output data rate).

The limitations identified suggest a series of follow-up studies and advancements, including:

- Testing the methodology on additional systems: Applying the MBSE and non-MBSE approaches on additional systems can help validate the methodology across additional engineering domains, as well as gather subsequent data on how MBSE approaches can help describe, develop, and evaluate complex systems.
- Testing the methodology with different MBSE languages and tools: Different languages possess unique ontologies with knowledge constructs that have the potential to capture more higher-levels of knowledge. Different modeling tools vary in their implementation of the MBSE languages, ability to generate extensions, and possession of unique, tool-specific features used to capture and organize system knowledge, automate information transfer, and automate knowledge processing.
- Testing the methodology with two independent development teams: This research was carried out by a single development team, which applied the MBSE and non-MBSE approaches in parallel. With this approach, there is a potential that decisions made for one

of the implementation approaches could have influenced the other. Testing the methodology with two independent development teams could help insure that the decision-making in the MBSE and non-MBSE approaches remain independent and decoupled.

- Developing and testing the extensive MBSE approach with increased automatic information transfer: The current MBSE approach used to architect the COM only utilized MBSE during the architecture definition tasks. The trade study and peer review tasks did not utilize MBSE and were still document-based. To address this issue, potential opportunities to apply MBSE methods to the trade study and peer review tasks were explored and proposed with an extensive MBSE approach. This extensive MBSE approach should be implemented on future system architecting activities to validate the approach and measure its ability to improve knowledge capture, increase automatic information transfer for the trade study and peer review tasks, and assist in architecture evaluation.
- Testing the modeling and simulation MBSE approach on a greater variety of requirements: Parameter-based requirements that are relatively straightforward to represent in models and make use of available functions built into the simulation tool (e.g., built-in rollup patterns in Cameo Systems Modeler) can be easily modeled, simulated, and verified through analysis in a virtual environment. Other requirements may not be easily modeled, simulated, and verified in a virtual environment, or may only be verifiable through inspection, demonstration, or test. Testing out the modeling and simulation process on a wider set of requirements will help both assess the extent of the

MBSE approach to verify system requirements, as well as understand its limitations in the types of requirements that can be modeled, simulated, and verified.

REFERENCES

[1] Way, D., “Preliminary Assessment of the Mars Science Laboratory Entry, Descent, and Landing Simulation,” IEEE Aerospace Conference, Big Sky, MT, Mar. 2-9, 2013.

[2] Grotzinger, J. P., J. Crisp, A. R. Vasavada, R. C. Anderson, C. J. Baker, R. Barry, D. F. Blake, P. Conrad, K. S. Edgett, B. Ferdowski, R. Gellert, J. B. Gilbert, M. Golombek, J. Gómez-Elvira, D. M. Hassler, L. Jandura, M. Litvak, P. Mahaffy, J. Maki, M. Meyer, M. C. Malin, I. Mitrofanov, J. J. Simmonds, D. Vaniman, R. V. Welch, and R. C. Wiens, “Mars Science Laboratory Mission and Science Investigation,” Space Sci. Rev., vol. 170, pp. 5-56, Sept. 2012, DOI: 10.1007/s11214-012-9892-2.

[3] Guizzo, E., "Planetary Rovers: Are We Alone?" IEEE Spectrum, Dec. 30, 2010.

[4] Smith, A., “How Many Jobs did the Mars Landing Create?” CNN Money, Aug. 6, 2012.

[5] United States Government Accountability Office, NASA: Assessments of Major Projects, GAO-12-207SP, Washington, DC, USA: United States Government Accountability Office, Mar. 2012.

[6] United States Government Accountability Office, NASA: Assessments of Selected Large-Scale Project, GAO-09-306SP, Washington, DC, USA: United States Government Accountability Office, Mar. 2009.

[7] Welch, R., D. Limonadi, and R. Manning, “Systems Engineering the Curiosity Rover: A Retrospective,” Proc. 8th International Conf. on System of Systems Engineering, Jun. 2-6, 2013, pp. 70-75, DOI: 10.1109/SYSoSE.2013.6575245.

[8] Mustard, J. F., M. Adler, A. Allwood, D. S. Bass, D. W. Beaty, J. F. Bell III, W. B. Brinckerhoff, M. Carr, D. J. Des Marais, B. Drake, K. S. Edgett, J. Eigenbrode, L. T. Elkins-Tanton, J. A. Grant, S. M. Milkovich, D. Ming, C. Moore, S. Murchie, T. C. Onstott, S. W. Ruff, M. A. Sephton, A. Steele, and A. Treiman, Report of the Mars 2020 Science Definition Team, Mars Exploration Program Analysis Group (MEPAG), 2013.

[9] Williford, K., K. Farley, K. Stack, A. Allwood, D. Beaty, L. Beegle, R. Bhartia, A. Brown, M. de la Torre Juarez, S. Hamran, M. Hecht, J. Hurowitz, J. Rodriguez-Manfredi, S. Maurice, S. Milkovich, R. Wiens, Roger, “The NASA Mars 2020 Rover Mission and the Search for Extraterrestrial Life,” Habitability to Life on Mars, pp. 275-308, 2018.

[10] National Aeronautics and Space Administration, FY2020 Explore Budget Estimates, NASA, 2019.

[11] Witze, A., "The $2.4-billion Plan to Steal a Rock from Mars," Nature, Jan. 18, 2017.

[12] United States Government Accountability Office, NASA: Assessments of Major Projects, GAO-19-262SP, Washington, DC, USA: United States Government Accountability Office, May 2019.

[13] United States Government Accountability Office, NASA: Assessments of Major Projects, GAO-21-306, Washington, DC, USA: United States Government Accountability Office, May 2021.

[14] United States Government Accountability Office, NASA: Assessments of Major Projects, GAO-17-303SP, Washington, DC, USA: United States Government Accountability Office, May 2017.

[15] Dvorak, D., “NASA Study on Flight Software Complexity,” NASA Office of Chief Engineer, 2009.

[16] Butler, R., and M. Pennotti, "The Evolution of Software and Its Impact on Complex System Design Robotic Spacecraft Embedded Systems," 2013 Conference on Systems Engineering Research, Atlanta, GA, Mar. 19-22, 2013.
[17] JPL 2018 Strategic Implementation Plan, Jet Propulsion Laboratory, California Institute of Technology, 2018.
[18] Zurbuchen, T., and D. Parker, "Joint Statement of Intent between the National Aeronautics and Space Administration and the European Space Agency on Mars Sample Return," April 26, 2018.
[19] National Academies of Sciences, Engineering, and Medicine, Visions into Voyages for Planetary Sciences in the Decade 2013-2022: A Midterm Review, Washington, DC, USA: The National Academies Press, 2018, DOI: 10.17226/25186.
[20] Muirhead, B., and A. Karp, "Mars Sample Return Lander Mission Concepts," Proc. IEEE Aero. Conf., Big Sky, MT, USA, Mar. 2-9, 2019, DOI: 10.1109/AERO.2019.8742215.
[21] Bayer, T., "Is MBSE helping? Measuring Value on Europa Clipper," Proc. IEEE Aero. Conf., Big Sky, MT, USA, Mar. 3-10, 2018, pp. 1-13, DOI: 10.1109/AERO.2018.8396379.
[22] Younse, P., C. Chiu, J. Cameron, M. Dolci, E. Elliot, A. Ishigo, D. Kogan, E. Marteau, V. Malyan, J. Mayo, J. Munger, P. Ohta, E. Olds, L. Strahle, and R. Willson, "Concept for an on-orbit Capture and Orientation Module for potential Mars Sample Return," Proc. IEEE Aero. Conf., Big Sky, MT, USA, Mar. 7-14, 2020, pp. 1-22, DOI: 10.1109/AERO47225.2020.9172814.
[23] Topper, J., and N. Horner, "Model-based Systems Engineering in Support of Complex Systems Development," John Hopkins APL Technical Digest, vol. 32, no. 1, pp. 419-432, 2013.
[24] Carroll, E., and R. Malins, Systematic Literature Review: How is Model-Based Systems Engineering Justified? Albuquerque, NM, USA: Sandia National Laboratories, 2016, DOI: 10.2172/1561164.
[25] Friedenthal, S., A. Moore, and R. Steiner, A Practical Guide to SysML, 3rd ed., Needham, MA, USA: OMG Press, 2014, DOI: 10.1016/C2013-0-14457-1.
[26] Madni, A., and S. Purohit, "Economic Analysis of Model-based Systems Engineering," Systems, vol. 7, no. 12, pp. 1-18, Feb. 2019, DOI: 10.3390/systems7010012.
[27] Chhaniyara, S., C. Saaj, M. Althoff-Kotzias, I. Ahrns, and B. Maediger, "Model Based Systems Engineering for Space Robotics Systems," 11th Symposium on Advanced Space Technologies in Robotics and Automation, ESTEC, Noordwijk, The Netherlands, Apr. 12-14, 2011.
[28] Rahman, M., and M. Mizukawa, "Model-Based Development and Simulation for Robotic Systems with SysML, Simulink and Simscape Profiles," International Journal of Advanced Robotic Systems, vol. 10, 112, 2013.
[29] No Magic, Cameo Systems Modeler User Guide, v. 18.1, No Magic, 2015.
[30] Borky, J., and T. Bradley, Effective Model-Based Systems Engineering, Cham, Switzerland: Springer, 2019, DOI: 10.1007/978-3-319-95669-5.
[31] Elaasar, M., N. Rouquette, S. Jenkins, and S. Gerard, "The Case for Integrated Model Centric Engineering," Proceedings of Model Based Enterprise Summit, Gaithersburg, MD, Apr. 2019.
[32] Haskins, C., "A Historical Perspective of MBSE with a View to the Future," INCOSE International Symposium, 21(1), 2011.

[33] Kim, S., D. Wagner, and A. Jimenez, "Challenges in Applying Model-Based Systems Engineering: Human-Centered Design Perspective," INCOSE Human Systems Integration Conference, Biarritz, France, Sept. 11-13, 2019.
[34] Huckaby, J., and H. Christensen, "A Case for SysML in Robotics," 2014 IEEE Conference on Automation Science and Engineering, Taipei, Taiwan, Aug. 18-22, 2014.
[35] Delligatti, L., SysML Distilled: A Brief Guide to the Systems Modeling Language, Upper Saddle River, NJ, USA: Addison-Wesley, 2014.
[36] Object Management Group, OMG Systems Modeling Language, v. 1.6, Needham, MA: Object Management Group, 2018.
[37] Object Management Group, OMG Unified Modeling Language, v. 2.5.1, Needham, MA: Object Management Group, 2017.
[38] Dori, D., Model-Based Systems Engineering with OPM and SysML, New York, NY: Springer, 2016.
[39] Modelica Association, Modelica - A Unified Object-Oriented Language for Systems Modeling, v. 3.4, 2017.
[40] Voirin, J., Model-based System and Architecture Engineering with the Arcadia Method, London, UK: ISTE Press Ltd., 2018.
[41] Long, D., and Z. Scott, A Primer for Model-Based Systems Engineering, 2nd ed., Blacksburg, VA, USA: Vitech Corporation, 2011.
[42] Hoffmann, H., Systems Engineering Best Practices with the Rational Solution for Systems and Software Engineering, Release 4.1, IBM Corporation, 2014.
[43] Weilkiens, T., SYSMOD – The Systems Modeling Toolbox, 2nd ed., MBSE4U, 2016.
[44] Aleksandraviciene, A., and A. Morkevicius, MagicGrid Book of Knowledge, Kaunas, Lithuania: Vitae Litera, 2018.
[45] Krob, D., CESAM: CESAMES Systems Architecting Method, Paris, France: C.E.S.A.M.E.S. Groupe, 2017.
[46] Huldt, T., and I. Stenius, "State-of-practice Survey of Model-based Systems Engineering," Systems Engineering, vol. 22, no. 2, Mar. 2019, DOI: 10.1002/sys.21466.
[47] Mazzini, S., E. Tronci, C. Paccagnini, and X. Olive, "A Model-based Methodology to Support the Space System Engineering (MBSSE)," Proc. ERTS2 2010, Embedded Real Time Software and Systems, Toulouse, France, May 19-21, 2010.
[48] IBM Corporation, Rational Rhapsody User Guide, v. 7.5, 2019.
[49] Roques, P., Systems Architecture Modeling with the Arcadia Method: A Practical Guide to Capella, London, UK: ISTE Press Ltd., 2017.
[50] Vitech Corporation, GENESYS System Definition Guide, v. 6.0, Blacksburg, VA: Vitech Corporation, 2018.
[51] Fernandez, J., and C. Hernandez, Practical Model-Based Systems Engineering, Norwood, MA, USA: Artech House, 2019.
[52] Murray, J., Model Based Systems Engineering (MBSE) Media Study, 2012.
[53] Herzig, S., R. Karban, G. Trancho, F. Dekens, N. Jankevicius, and M. Troy, "Analyzing the Operational Behavior of the Alignment and Phasing System of the Thirty Meter Telescope using SysML," Adaptive Optics for Extremely Large Telescopes Proceedings, Canary Islands, Spain, Jun. 25-30, 2017.
[54] White, C., and B. Mesmer, "Research Needs in Systems Engineering: Report from a University of Alabama in Huntsville workshop," Systems Engineering, vol. 23, no. 2, pp. 154-164, Mar. 2020, DOI: 10.1002/sys.21501.

[56] Mcginnis, L., and V. Ustun, "A Simple Example of SysML-driven Simulation," Proceedings of the 2009 Winter Simulation Conference, Austin, TX, Dec. 13-16, 2009.
[57] Karban, R., N. Jankevicius, and M. Elaasar, "ESEM: Automated Systems Analysis using Executable SysML Modeling Patterns," INCOSE International Symposium (IS), Edinburgh, Scotland, 2016.
[58] Fosse, E., and C. Harmon, "Inheriting Curiosity: Leveraging MBSE to Build Mars 2020," AIAA SPACE 2015 Conference and Exposition, Pasadena, CA, Aug. 31 – Sept. 2, 2015.
[59] Sindiy, O., T. Mozafari, and C. Budney, "Application of Model-Based Systems Engineering for the Development of the Asteroid Redirect Robotic Mission," AIAA SPACE Forum, Long Beach, CA, Sept. 13-16, 2016.
[60] Wibben, D., and R. Furfaro, "Model-Based Systems Engineering Approach for the Development of the Science Processing, and Operations Center of the NASA OSIRIS-Rex Asteroid Sample Return Mission," Acta Astronautica, 115, 2015.
[61] Estable, S., T. Granger, T. Lochow, T. Zoebelein, N. Brauer, I. Tolchinsky, S. Gerene, and H. Koning, "Systems Modelling and Simulation of the ESA e.Deorbit Space Debris Removal Mission," Proc. NAFEMs, Stockholm, Sweden, Jan. 20, 2017.
[62] No Magic, Cameo Simulation Toolkit, v. 18.1, No Magic, 2014.
[63] Kaslow, D., G. Soremekun, H. Kim, and S. Spangelo, "Integrated Model-based Systems Engineering (MBSE) Applied to the Simulation of a CubeSat Mission," 2014 IEEE Aerospace Conference, Big Sky, MT, Mar. 1-8, 2014.
[64] Gregory, J., L. Berthoud, T. Tryfonas, and A. Prezzavento, "Early Validation of the Data Handling Unit of a Spacecraft Using MBSE," 2019 IEEE Aerospace Conference, Big Sky, MT, Mar. 2-9, 2019.
[65] Wolny, S., A. Mazak, C. Carpella, V. Geist, and M. Wimmer, "Thirteen Years of SysML: A Systematic Mapping Study," Software & Systems Modeling, 2019.
[66] Rogers, E., Diffusion of Innovations, 5th ed., New York, NY: The Free Press, 2003.
[67] Wang, Lui, M. Izygon, S. Okon, L. Garner, and H. Wagner, "Effort to Accelerate MBSE Adoption and Usage at JSC," AIAA SPACE 2016, Long Beach, CA, Sep. 13-16, 2016.
[68] Chami, M., A. Morkevicius, A. Alkeksandravicience, and J. Bruel, "Towards Solving MBSE Adoption Challenges: The D3 MBSE Adoption Toolbox," 28th Annual INCOSE International Symposium, Washington, DC, Jul. 7-12, 2018.
[69] Trase, K., and E. Fink, "A Model-Driven Visualization Tool for Use with Model-Based Systems Engineering Projects," 2014 IEEE Aerospace Conference, Big Sky, MT, Mar. 1-8, 2014.
[70] Fu, W., and W. Gray, "Resolving the Paradox of the Active User: Stable Suboptimal Performance in Interactive Tasks," Cognitive Sciences, 28(6), 901-935, 2004.
[71] Kornfeld, R., J. Parrish, S. Sell, and S. May, "Mars Sample Return: Testing the Last Meter of Rendezvous and Sample Capture," Journal of Spacecraft and Rockets, 44(3), Jun. 2007.
[72] Mukherjee, R., B. Chamberlain-Simon, R. Smith, M. Dolci, R. McCormick, and P. Ohta, "Concepts for Mars On-Orbit Robotic Sample Capture and Transfer," IEEE Aerospace Conference, Big Sky, MT, Mar. 4-11, 2017.
[73] Parrish, J., "To Mars and Back: Technologies for a Potential NASA Mars Sample Return," Mountain View, CA, Jun. 14, 2017.
[74] Younse, P., J. Strahle, M. Dolci, P. Ohta, K. Lalla, and E. Olds, "An Orbiting Sample Capture and Orientation Element Architecture for Potential Mars Sample Return," 2018 IEEE Aerospace Conference, Big Sky, MT, Mar. 3-10, 2018.

[75] Younse, P., R. Adajian, B. Cano, M. Dolci, P. Ohta, K. Lalla, V. Malyan, J. Munger, E. Olds, and J. Strahle, "Sample Capture and Orientation Technologies for Potential Mars Sample Return," Proc. 2nd International Mars Sample Return, 6027, Apr. 25, 2018.
[76] National Aeronautics and Space Administration, NASA Technology Roadmaps, Introduction, Crosscutting Technologies, and Index, Washington, DC: NASA Systems Engineering Handbook, 2015.
[77] National Aeronautics and Space Administration, NASA Systems Engineering Handbook, SP-2016-6105 Rev2, Washington, DC, USA: National Aeronautics and Space Administration, 2016.
[78] National Aeronautics and Space Administration, NASA Systems Engineering Processes and Requirements, NPR 7123.1, Washington, DC, USA: National Aeronautics and Space Administration, 2013.
[79] Pohl, K., Requirements Engineering, Berlin, Germany: Springer, 2010.
[80] Hood, C., S. Wiedemann, S. Fichtinger, and U. Pautz, Requirements Management: The Interface Between Requirements Development and All Other Systems Engineering Processes, Oberhaching, Germany: Springer, 2008.
[81] National Aeronautics and Space Administration, NASA Space Flight Program and Project Management Handbook, NASA SP-2014-3705, Washington, DC: NASA Systems Engineering Handbook, 2014
[82] Lambert, J., and C. Frye, Microsoft Office 2019 Step by Step, Redmond, WA: Microsoft Press, 2018.
[83] IBM Corporation, "Ditch the Documents and Spreadsheets – Manage Requirements Efficiently and More Accurately with IBM Rational DOORS Next Generation," Somers, NY: IBM Corporation, 2014.
[84] Siemens Product Lifecycle Management Software Inc., Getting Started with Teamcenter, PLM00002 11.2.2, Plano, TX: Siemens Industry Software, 2016.
[85] Atlassian, Documentation for Confluence 7.4, Sydney, Australia: Atlassian, 2020.
[86] Roques, P., "MBSE with the ARCADIA Method and the Capella Tool," 8th European Congress on Embedded Real Time Software and Systems, Toulouse, France, Jan. 27-29, 2016.
[87] Yang, G., Contemporary Planetary Robotics, Weinheim, Germany: Wiley-VCH, 2016.
[88] Blockley, R., and W. Shyy, Encyclopedia of Aerospace Engineering, Hoboken, NJ: John Wiley & Sons, Inc., 2010.
[89] ISO, Systems and Software Engineering – Architecture Description, ISO/IEC/IEEE 42010, 2011.
[90] Crawley, E., B. Cameron, and D. Selva, Systems Architecture: Strategy and Product Development for Complex Systems, Hoboken, NJ: Pearson Higher Education, Inc., 2016.
[91] Weilkiens, T., J. Lamm, S. Roth, and M. Walker, Model-Based System Architecture, Hoboken, NJ: John Wiley & Sons, Inc., 2016.
[92] INCOSE, Systems Engineering Handbook, 4th ed., San Diego, CA: INCOSE, 2015.
[93] Min, I., and R. Noguchi, "The Architecture Design and Evaluation Process: A Decision Support Framework for Conducting and Evaluating Architecture Studies," IEEE Aerospace Conference, Big Sky, MT, Mar. 5-12, 2016.
[94] Raz, A., C. Kenley, and D. DeLaurentis, "System Architecting and Design Space Characterization," Systems Engineering, 21:227-242, 2018.

[95] Emery, D., and R. Hilliard, "Every Architecture Description Needs a Framework: Expressing Architecture Frameworks using ISO/IEC 42010," Proc. Joint Working IEEE/IFIP Conf. on Software Architecture & European Conference on Software Architecture, Cambridge, UK, Sept. 14, 2009, pp. 31-40, DOI: 10.1109/WICSA.2009.5290789.
[96] The Open Group, The TOGAF Standard, v. 9.2, San Francisco, CA, 2018.
[97] United States Department of Defense Office, Chief Information Officer, DoD Architecture Framework, v. 2.02, Change 1, 2015.
[98] Blokyk, G., Zachman Framework A Complete Guide – 2021 Edition, The Art of Service - Zachman Framework Publishing, 2020.
[99] Maier, M. W., and E. Rechtin, The Art of Systems Architecting, Boca Raton, FL: CRC Press, 2009.
[100] McKelvin, M. L., R. Castillo, K. Bonanne, M. Bonnici, B. Cox, C. Gibson, J. P. Leon, J. Gomez-Mustafa, A. Jimenez, and A. Madni, "A Principled Approach to the Specification of System Architectures for Space Missions," AIAA SPACE 2015 Conference and Exposition, Pasadena, CA, Aug. 31-Sep. 2, 2015.
[101] Ryschkewitsch, M., D. Schaible, and W. Larson, "The Art and Science of Systems Engineering," Systems Research Forum, Vol. 03, No. 02, pp. 81-100, 2009.
[102] Anderson, L. (Ed.), D. Krathwohl (Ed.), P. Airasian, K. Cruikshank, R. Mayer, P. Pintrich, J. Raths, and M. Wittrock, A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Complete ed. New York, NY: Addison Wesley Longman, Inc., 2001.
[103] Jet Propulsion Laboratory, JPL Design Principles, Rev. 8, Pasadena, CA: California Institute of Technology, 2020.
[104] Oberhettinger, D., "Assuring that Lessons Learned Critical to Mission Success Get Used," IEEE Aerospace Conference, Big Sky, MT, Mar. 3-12, 2012.
[105] National Aeronautics and Space Administration, "Galileo Scan Platform Anomaly at Ida Encounter," NASA Lessons Learned Entry 560, NASA Engineering Network, Washington, DC: NASA Systems Engineering Handbook, 1997.
[106] National Aeronautics and Space Administration, "MSL Sol-200 Anomaly," NASA Lessons Learned Entry 11201, NASA Engineering Network, Washington, DC: NASA Systems Engineering Handbook, 1997.
[107] Vincenti, W., What Engineers Know and How They Know It, Baltimore, MD: John Hopkins University Press, 1990.
[108] Ellery, A., Planetary Rovers: Robotic Space Exploration of the Solar System, Chichester, UK: Praxis Publishing, 2016.
[109] Murugesan, S., "An Overview of Electric Motors for Space Applications," IEEE Transactions on Industrial Electronics and Control Instrumentation, vol. IECI-28, no. 4, Nov. 1981.
[110] Midwest Research Institute, Brushless DC Motors, NASA CR-2506, Washington, DC: National Aeronautics and Space Administration, 1975.
[111] National Aeronautics and Space Administration, Expanded Guidance for NASA Systems Engineering, SP-2016-6105-SUPPL, Washington, DC, USA: National Aeronautics and Space Administration, 2016.
[112] Fortescue, P., G. Swinerd, and J. Stark, Spacecraft Systems Engineering, 4th ed., West Sussex, UK: John Wiley & Sons, Ltd., 2011.

[113] Herzig, S., D. Velez, B. Nairouz, B. Weatherspoon, R. Tikidjian, T. Randolph, and B. Muirhead, “A Model-based Approach to Developing the Concept of Operations for Potential Mars Sample Return,” Proc. AIAA SPACE and Astronautics Forum and Exposition, Orlando, FL, USA, Sept. 17-19, 2018, pp. 1-15, DOI: 10.2514/6.2018-5389.
[114] Huesing, J., F. Beyer, H. P. de Koning, and J. Delfa, “MBSE for MSR – Introducing MBSE to Early Phase Mission Design for Mars Sample Return,” Proc. SECESA, Glasgow, UK, Sept. 26-28, 2018.
[115] Ryschkewitsch, M., D. Schaible, and W. Larson, “The Art and Science of Systems Engineering,” Systems Research Forum, Vol. 3, No. 2, 2009.
[116] Way, D., “Preliminary Assessment of the Mars Science Laboratory Entry, Descent, and Landing Simulation,” Proc. IEEE Aero. Conf., Big Sky, MT, USA, Mar. 2-9, 2013, pp. 1-16., DOI: 10.1109/AERO.2013.6497404.
[117] Muirhead, B., A. Nicholas, and J. Umland, “Mars Sample Return Mission Concept Status,” Proc. IEEE Aero. Conf., Big Sky, MT, USA, Mar. 7-14, 2020, pp. 1-8, DOI: 10.1109/AERO47225.2020.9172609.
[118] National Aeronautics and Space Administration, Mars Sample Return Program Final Report of the Independent Review Board, Washington, DC, USA: National Aeronautics and Space Administration, 2020.
[119] Eppinger, S., and T. Browning, Design Structure Matrix Methods and Applications, Cambridge, MA, USA: MIT Press, 2012, DOI: 10.7551/mitpress/8896.001.0001.
[120] Henderson, K., and A. Salado, "Value and Benefits of Model-based Systems Engineering (MBSE): Evidence from the Literature," Systems Engineering, vol. 24, no. 1, pp. 51-66, Jan. 2021, DOI: 10.1002/sys.21566.
[121] Rashid, M., M. Anwar, and A. Khan, "Toward the Tools Selection in Model Based System Engineering for Embedded Systems—A systematic Literature Review," Journal of Systems and Software, vol. 106, pp. 150-163, Aug. 2015, DOI: 10.1016/j.jss.2015.04.089.
[122] Estefan, J., “Survey of Model-based Systems Engineering (MBSE) Methodologies,” INCOSE MBSE Focus Group, vol. 25, pp. 1-12, 2007.
[123] Madni, A., and M. Sievers, “Model-based Systems Engineering: Motivation, Current Status, and Research Opportunities,” Systems Engineering, vol. 170, no. 3, pp. 172-190, May 2018, DOI: 10.1002/sys.21438.
[124] BKCASE Editorial Board, The Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.4, Hoboken, NJ, USA: The Trustees of the Stevens Institute of Technology, 2021.
[125] Systems and Software Engineering – Architecture Description, ISO/IEC/IEEE 42010, 2011.
[126] Cordero, S., C. Fortin, and R. Vingerhoeds, “Concurrent Conceptual Design Sequencing for MBSE of Complex Systems through Design Structure Matrices,” Proc. Design Society: DESIGN Conf., vol. 1, Cavtat, Croatia, Oct. 26-29, 2020, pp. 2375-2384, DOI: 10.1017/dsd.2020.96.
[127] Saunders, S., “Promoting the Real Value of Systems Engineering using an Extended SCARIT Process Model,” Proc. INCOSE International Symposium, vol. 17, no. 1, San Diego, CA, USA, Jun. 24-28, 2007, DOI: 10.1002/j.2334-5837.2007.tb02893.x.
[128] Younse, P., J. Strahle, M. Dolci, P. Ohta, K. Lalla, and R. Adajian, “A Systems Architecting Methodology using Bloom’s Taxonomy to Promote Creative Engineering

Synthesis," Proc. IEEE Aero. Conf., Big Sky, MT, USA, Mar. 3-10, 2018, pp. 1-24, DOI: 10.1109/AERO.2018.8396453.
[129] McDermott, T., N. Hutchison, M. Clifford, E. Van Aken, A. Salado, and K. Henderson, Benchmarking the Benefits and Current Maturity of Model-Based Systems Engineering across the Enterprise, Technical Report SERC-2020-SR-001, Hoboken, NJ, USA: Stevens Institute of Technology, Systems Engineering Research Center, 2020.
[130] Carlson, R., and W. Vaneman, "Evaluating Current Systems Engineering Models for Applicability to Model-based Systems Engineering Technical Reviews," Naval Engineers Journal, vol. 132, no. 2, Jun. 2020.
[131] Younse P., J. Cameron, and T. Bradley. "Comparative Analysis of a Model-based Systems Engineering Approach to a Traditional Systems Engineering Approach for Architecting a Robotic Space System through Knowledge Categorization," Systems Engineering, 2021.
[132] Ishigo, A., E. Mareau, P. Ohta, E. Elliot, and P. Younse, "Modeling, Simulation, and Analysis of Orbiting Sample Capture for Potential Mars Sample Return," IEEE Aerospace Conference, Big Sky, MT, Mar. 7-14, 2020.
[133] Strahle, J., C. Chiu, M. Dolci, and P. Younse, "Robotics System Process and Concept for On-orbit Assembly for Potential Mars Sample Return," IEEE Aerospace Conference, Big Sky, MT, Mar. 7-14, 2020.
[134] National Science Foundation (NSF) Blue Ribbon Panel Report on Simulation-Based Engineering Science: Revolutionizing Engineering Science through Simulation, NSF Press, 2006.
[135] Zeigler, B. P., S. Mittal, and M. K. Traore, "MBSE with/out Simulation: State of the Art and Way Forward," Systems, 2018.
[136] Karban R., F. Dekens, S. Herzig, M. Elaasar, and N. Jankevičius, "Creating System Engineering Products with Executable Models in a Model-based Engineering Environment," Modeling, Systems Engineering, and Project Management for Astronomy, 2016.
[137] Henderson, K., and A. Salado, "Value and Benefits of Model-based Systems Engineering (MBSE): Evidence from the Literature," Systems Engineering, 2021.
[138] Department of Defense Directive, "DoD Modeling and Simulation (M&S) Management," No. 5000.59, 2007.
[139] Loper, M. L., Modeling and Simulation in the Systems Engineering Life Cycle: Core Concepts and Accompanying Lectures, Springer, 2015.
[140] Reilley K. A., S. Edwards, R. Peak, and D. Mavris, "Methodologies for Modeling and Simulation in Model-based Systems Engineering Tools," AIAA SPACE, 2016.
[141] Jinzhi, L., "Research Survey on MBSE Tool-chain," School of Industrial Engineering and Management Reading Reports, 2019.
[142] Gregory, J., L. Berthoud, T. Tryfonas, A. Prezzavento, and L. Faure "Investigating the Flexibility of the MBSE Approach to the Biomass Mission," IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2020.
[143] Walker, L., and L. Thomas, "Integrated System Modeling in SysML for Small Satellites," AIAA Modeling and Simulation Technologies Conference, Grapevine, TX, Jan. 9-13, 2017.
[144] Charpigny, N., "An Executable System Model for Behavioural Analyses of the LISA Mission," KTH Royal Institute of Technology School of Engineering Sciences, TRITA-SCI-GRU, 2019.
[145] Haskins C., K. Forsberg, M. Krueger, D. Walden, and D. Hamelin, Systems Engineering Handbook, INCOSE, 2006.

[146] Gräßler, I., J. Hentze, and T. Bruckmann, "V-models for Interdisciplinary Systems Engineering," Proceedings of the DESIGN 2018 15th International Design Conference, 2018.
[147] Obstbaum, M., U. Wurstbauer, C. König, T. Wagner, C. Kübler, and V. Fäßler, “From a Graph to a Development Cycle: MBSE as an Approach to Reduce Development Efforts,” Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth, 2017.
[148] Hatakeyama, S., D. Seal, D. Farr, and S. Haase, “An Alternate View of the Systems Engineering "V" in a Model-Based Engineering Environment,” 2018 AIAA SPAE and Astronautics Forum and Exposition, Orlando, FL, Sept. 17-19, 2018.
[149] Eigner, M., T. Gilz, and R. Zafirov, “Interdisciplinary Product Development-Model Based Systems Engineering,” PML Portal, 2012.
[150] Sargent, R., “An Assessment Procedure and a Set of Criteria for Use in the Evaluation of Computerized Models and Computer-Based Modeling Tools,” Final Technical Report RADC-TR-80-409, U.S. Air Force, 1981.
[151] Sargent, R., “Verification and Validation of Simulation Models,” Winter Simulation Conference, Dec. 11-14, 2011.
[152] Carson, J., “Model Verification and Validation,” Winter Simulation Conference, Dec. 8-11, 2002.
[153] Honour, E., Systems engineering return on investment, Adelaide: University of South Australia, 2013.
[154] Trujillo, L., "Mental Effort and Information-Processing Costs Are Inversely Related to Global Brain Free Energy During Visual Categorization," Frontiers in Neuroscience, 2019.
[155] Sweller, J., J. Merrienboer, and F. Paas, "Cognitive architecture and instructional design," Educational Psychology Review, 1998.
[156] Eppler, M., and J. Mengis. "The Concept of Information Overload - A Review of Literature from Organization Science, Accounting, Marketing, MIS, and Related Disciplines," Kommunikationsmanagement im Wandel, 2008.