MODELING ARCHITECTURES AND PARAMETERIZATION OF SPACECRAFT WITH APPLICATION TO PERSISTENT PLATFORMS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Melanie L. Grande

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

August 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Daniel A. DeLaurentis, Chair

School of Aeronautics and Astronautics

Dr. C. Robert Kenley

School of Industrial Engineering

Dr. Kathleen C. Howell

School of Aeronautics and Astronautics

Dr. Cesare Guariniello

School of Aeronautics and Astronautics

Approved by:

Dr. Gregory Blaisdell

Head of Gambaro Graduate Program of Aeronautics and Astronautics

ACKNOWLEDGMENTS

I would like to thank my adviser, Dr. Dan DeLaurentis, for his guidance and support. Thank you for encouraging me to follow my passion to explore new concepts in space exploration. I would also like to thank Dr. Cesare Guariniello, for encouraging me through his own passion for space, which must surely be twice my own. Thank you for introducing me to the SoS Lab and for always offering a cup of tea and a smile.

To my committee members, Dr. Bob Kenley and Dr. Kathleen Howell: thank you for supporting my research. I am grateful for all the stimulating questions and guidance that helped me broaden my perspective. To the members of the SoS Lab: thank you for being a welcoming and supportive group.

Finally, I would like to thank my family and the friends who supported me throughout my years at Purdue. Thank you for sharing with me the vacations to a sunnier, warmer home state, weekend climbing trips, board games on weekends, companions to go with on thrilling travels around the world, and all the fun memories that balanced out the difficult times.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

SYMBOLS

i	index of value measures
j	index of alternatives
w_i	swing weight for ith value measure
x_{ji}	jth alternative's score for ith value measure
$v_i(x_{ji})$	normalized value of the score x_{ji}
v_j	total value for jth alternative
t	architecture design alternative counter
A	number of assemblies in an architecture
a	assembly counter
M	number of modules in an assembly
m	module counter
N	number of subsystems in a module
n	subsystem counter
J_m	adjacency matrix of module m
G_m	module matrix of module m
$H_a(t)$	assembly matrix of assembly a at time t
TD	total number of design alternatives
FD	number of feasible and acceptable design alternatives
F	number of feedback loops
f	feedback loop counter
i	link counter for links in loops
W_{i-f}	weight of link i in loop f
l_f	number of links in loop f
k	link counter for links not in a loop
W_k	weight of link k

l_k	number of links not in a loop
L	number of links in a subsystem
C_{I_n}	integration complexity of subsystem n
α_I	coefficient of integration
C_{CC}	coupling complexity of a module
C_{CC_n}	coupling complexity of subsystem n
C_{MC_m}	module complexity of module m
C_{A_t}	architecture complexity of architecture t
F_t	flexibility of architecture t
P_t	performance of architecture t
ΔP_t	change in performance of architecture t
NP_t	normalized performance change of architecture t
DC_{r-t}	design changes from reference to architecture t
MRP	mass robustness penalty
K	performance comparison constant
C_{DP}	total cost of DDT&E and production
C_{ops}	annual cost of operations
C_{NR}	non-recurring cost
C_R	recurring cost
Q	quantity
M	dry mass
S	specification
IOC	year of initial operation capability
B	block generation
D	difficulty
P	power
DR	data rate

ABBREVIATIONS

AMCM	Advanced Mission Cost Model
ABM	Agent-Based Modeling
CAR	control accuracy requirement
CER	cost estimating relationship
COPUOS	Committee on the Peaceful Uses of Outer Space
COTS	commercial-off-the-shelf
DARPA	Defense Advanced Research Projects Agency
DDT&E	Design Development Testing and Engineering
ESA	European Space Agency
EXAMINE	Exploration Architecture Model for IN-space and Earth-to-orbit
GINA	Generalized Information Network Analysis
GMAT	General Mission Analysis Tool
GT-FAST	GeorgiaTech F6 Architecture Synthesis Tool
IMLEO	initial mass in low-Earth orbit
IMU	intertial measurement unit
INCOSE	International Council on Systems Engineering
LEO	low-Earth orbit
MAPS	Modeling Architectures and Parameterization of Spacecraft
MATE	Multi-Attribute Tradespace Exploration
MAUA	Multi-Attribute Utility Analysis
MIT	Massachussetts Institute of Technology
MLI	Multi-Layer Insulation
NASA	National Aeronautics and Space Administration
NextSTEP	Next Space Technologies for Exploration Partnerships
NICM	NASA Instrument Cost Model

OOP	Object-Oriented Programming
OSAM	On-orbit Servicing, Assembly, and Maintenance
OTA	Orbital Trajectory Analysis
PCU	power control unit
SCM	system complicatedness measure
SE	systems engineering
SMAD	Space Mission Analysis and Design
SME	Space Mission Engineering
SoS	system-of-systems
SoSE	system-of-systems engineering
UN	United Nations
VCA	Value Centric Analysis
VCD	Value-Centric Design
VCDM	Value-Centric Design Methodology
VDD	Value-Driven Design

ABSTRACT

Grande, Melanie L. M.S., Purdue University, August 2020. Modeling Architectures and Parameterization of Spacecraft with Application to Persistent Platforms. Major Professor: Daniel A. DeLaurentis.

With the increasing entrance of diverse new ventures to today's space industry, there is a need for a dialogue and examples for advanced concepts to be approached with system-of-systems engineering (SoSE) methods. Since NASA and international partners have set their sights on returning humans to the Moon in the 2020s, there has been significant discussion of the potential for exploration and science missions. Additionally, in our existing Earth-orbit economy, servicing satellites and manufacturing technology have been demonstrated by both public and private actors. On-orbit servicing, assembly, and manufacturing (OSAM) stands to be the next "game changing" market in space.

A future OSAM market may especially benefit from SoSE methods. Many sources in literature as well as actors in both the industry and federal agencies have spoken about the value and game-changing nature of OSAM capabilities for the future of larger, longer-life, and more flexible space assets. However, there has been little discussion or effort so far to approach the design of OSAM ventures using SoSE. The SoSE concept definition methods and modeling framework are essential for identifying the multi-faceted resources, operations, policies, and economics, as well as stakeholders, disruptors, and drivers that impact an SoS. This thesis will provide an overview of the SoSE discipline as well as Value-Centric Analysis (VCA), which was the selected method for the SoSE modeling and analysis process. This thesis will also present a new modeling and analysis environment. The Modeling Architectures and Parameterization of Spacecraft (MAPS) environment was created specifically to respond to today's advanced space systems problems and to integrate existing SoSE and VCA

tools. Finally, a case study is presented to demonstrate the capabilities of MAPS. In the case study, the design characteristics of a persistent platform in orbit with OSAM capabilities are evaluated to provide decision support.

The methods presented in this thesis follow the three-phase process for SoSE: Definition, Abstraction, and Implementation. It begins with problem definition, then identification of stakeholders and value measures, and then proceeds to value and cost analysis using the MAPS environment. Each of the methods are applied to the case study of a persistent platform in low-Earth orbit. In the case study, over 18,000 architectures were analyzed to determine the design characteristics that best balanced value and cost.

Several strategies for demonstrating OSAM technologies in orbit have been presented to the public over time, but this thesis has defined physical architecture design options and relevant value measures. In addition, the full SoSE process was applied to an OSAM concept for the first time. This work is therefore a significant step towards providing future insight to decision makers. The dialogue on these topics and the SoSE methods should be valuable to a future OSAM market and other new ventures in the space industry.

CHAPTER 1. INTRODUCTION

As NASA and its commercial and international partners have set their sights on returning humans to the surface of the Moon in the 2020s, there has been significant discussion of the potential for exploration and science in cislunar space, including the orbital region of the Earth and moon. NASA has in addition been directed to kick-start a new commercial cislunar economy and leverage more public-private partnerships (NASA, 2019a),(NASA, 2019b). In our existing Earth-orbit economy, however, new space-based markets are still emerging. Existing markets for satellite-based communications and Earth observation continue to grow, but both the promises and first moves of new markets are emerging to support the other markets as well as create new opportunities. Included in these new movers are on-orbit servicing, assembly, and manufacturing (OSAM) concepts, with supporters that proclaim there is high potential value. Recently, some servicing satellites and manufacturing technology have been demonstrated by both public and private actors.

The above activities are all part of a complex industry web of engineers and manufacturers, primary and secondary contractors, operators and suppliers, public and private investors, visionaries and regulators. Therefore, it is important to consider both the opportunities in cislunar space and for OSAM in a way in which big-picture dynamics can be understood. Due to the space industry's complexity, it would be worthwhile to consider any concepts as a system-of-systems (SoS). This is similar to the foundation already laid by DeLaurentis, Sindiy, and Stein for using SoS engineering (SoSE) to understand, model, and analyze the U.S. National Space Program (DeLaurentis et al., 2006).

This chapter will introduce the two realms of opportunities—for markets in cislunar space and in OSAM—by employing SoS problem definition techniques. This chapter will also introduce specific research objectives for assessment of OSAM oppor-

tunities for this thesis. The first products of the Definition phase are identification of the operational context, status quo, and barriers (DeLaurentis et al., 2006), so these three will be discussed for each realm of opportunity. This technique is similar to others in literature, for example the method of "environmental scanning" where categories have been defined for the external environment—including social, regulatory, technological, political, economic, and industry—and the potential impact of each category is assessed for a proposed system or organization (Albright, 2004). In comparison, the categories presented by (DeLaurentis et al., 2006) are based on the impact to the new system rather than subsets of the environment, although it is important to consider different environments when identifying the operational context, status quo, and barriers.

1.1 Opportunities in Cislunar Space

To understand the opportunities in cislunar space as an SoS problem, the following sections will describe the products of the first phase of SoSE, Problem Definition, including operational context, status quo, and barriers towards realizing any new opportunities (DeLaurentis et al., 2006). It is also important to understand why cislunar space is a focus of discussions today. In approximately the last decade, there has been an emergence of many new players in the space industry, and the exploration landscape has therefore substantially changed since humans last set foot on the Moon in the 1970s, over four decades ago. The intersection of a new landscape and bold visions will create a wide variety of opportunities in cislunar space.

1.1.1 Operational Context

The Moon once again became the focus of American space exploration when Space Policy Directive 1 was signed by President Donald Trump on December 11, 2017 (Wang, 2017). Led by Administrator Jim Bridenstine, NASA then committed to the Artemis program to return humans to cislunar space and also committed to developing

a future cislunar commercial economy. However, several other countries have also set their sights on the moon. China has built ever greater capabilities in space as it makes its way ultimately towards a sustainable science base on the lunar surface (Bartels, 2019). The Chinese Chang'e Program began with the Chang'e-1 probe, launched to lunar orbit in October 2007, and most recently has made history with Chang'e-4, the first rover landed on the far side of the Moon in January 2019 (Mann, 2019). Russia announced in September 2019 that Roscosmos would partner with China, where the countries' orbiters and landers would work together and human missions could happen by 2030 (Bartels, 2019). The European Space Agency (ESA) has announced their focus on strategies for both performing science (European Space Agency, 2019b) and utilizing resources on the lunar surface during 2020 to 2030 (European Space Agency, 2019a). The ESA member states are also included as international partners in NASA's vision for cislunar space, together with American commercial partners (NASA, 2019b). In addition, various other countries worldwide have also become involved in the LEO economy over the last decade, impacting the space industry through advanced satellite technologies, manufacturing, and operations.

In the United States, the return to the Moon after so many decades was envisioned to be significantly different from the 1960s: increased public-private partnerships, an emphasis on sustainability, and support for the creation of a commercial cislunar economy. NASA has taken many steps towards realizing this vision; however, challenges do remain. With the support and direction of Congress, NASA has planned for commercial partners to be heavily involved in the Artemis program (NASA, 2019c). For example, the development of advanced human exploration elements has begun through a public-private partnership model called Next Space Technologies for Exploration Partnerships (NextSTEP) (NASA, 2019b). In fact, the original NextSTEP Omnibus was issued in 2014, predating the Artemis program and intended to guide the development of a cislunar Gateway. The Gateway—which first became public in 2012 as the "Deep Space Habitat"—is a NASA plan for a reusable command and ser-

vice station in lunar orbit, meant to serve as the first stepping stone in a sustainable return to the Moon (NASA, 2019c).

Then NASA was directed in March 2019 to return humans to the Moon by 2024—an ambitious schedule. Gateway was later removed from the critical path in March 2020, in short to "de-risk" the Artemis program by removing risks of cost and schedule overruns (Foust, 2020). It is clear that challenges remain for decision-makers to effectively develop a national space program that accounts for technology, infrastructure, operations, and policy together and that maximizes benefit over multiple generations while minimizing the impact of changing space program objectives—just as these challenges existed in the transition from the Space Transportation System era to the Constellation era circa 2005 (DeLaurentis et al., 2006).

Other commercial partnerships have also been established. NASA selected nine partners for Commercial Lunar Payload Services (CLPS) contracts in November 2018 and then released its first lunar surface delivery task order in March 2019 (NASA, 2019a). NextSTEP and CLPS are just two means that NASA is seeking public-private partnerships to fund and support advanced exploration capabilities. In addition, NASA envisions that commercial involvement can grow beyond government contracts. NASA officials have spoken about and begun to establish programs that might grow a new cislunar economy similarly to the existing LEO economy, with new markets and bold new investors. This may be an ambitious vision, as there are a long list of challenges and uncertainties associated with feasible missions to cislunar space and beyond. The feasibility of a business case in cislunar space–beyond what is directly supported by government contracts—has yet to be realized.

1.1.2 Status Quo

The status quo for cislunar activities is defined by the political, regulatory, and economic environments just as much as the technological environment. It is these environments which shape the drivers and barriers to any future mission or business

venture in cislunar space. The political and regulatory environment in the space industry today is marked by an increasing number of global actors in space and a growing number of countries and companies that have launched missions to the moon (Mann, 2019), (European Space Agency, 2019b), (Wall, 2019). However, the international governance of space activities remains somewhat ambiguous, with direction coming from a limited number of non-legally binding international agreements—namely, the "Outer Space Treaty" of 1967 (formally named the Treaty on Principles Governing the Activities of States in the Exploration and Use of Outer Space, including the Moon and Other Celestial Bodies). Follow-ups to the Outer Space Treaty included the Space Liability Convention of 1972 (Wikipedia, 2020) and the Registration Convention of 1976 (Wikipedia, 2019). The principles of this Treaty most relevant to envisioned activities in cislunar space include (United Nations Office for Outer Space Affairs, nd):

- "outer space shall be free for exploration and use by all States;"
- "outer space is not subject to national appropriation by claim of sovereignty, by means of use or occupation, or by any other means;"
- "States shall be responsible for national space activities whether carried out by governmental or non-governmental entities;"
- "States shall be liable for damage caused by their space objects; and"
- "States shall avoid harmful contamination of space and celestial bodies."

These principles should be kept in mind when envisioning a new venture to cislunar space, in the case they are restrictive for any reason. For example, territorial claims are prohibited (United Nations Office for Outer Space Affairs, nd), though the establishment of commercial operations on the lunar surface could be interpreted as territorial. Also, commercial ventures are to be held responsible for operations and contamination by their nation state (United Nations Office for Outer Space Affairs,

nd); however, states may not choose to or know how exactly to enforce the Treaty, whether the state has long been a player in outer space operations or they are newly developing space capabilities and policies.

On the other hand, the Treaty does not specifically regulate the usage of space resources, although it does proclaim that outer space is "free for exploration and use" (Article I) and "not subject to national appropriation" (Article III) (United Nations Office for Outer Space Affairs, nd). This is important because *in situ* resource utilization (ISRU) is a commonly studied means of establishing a sustainable base in deep space. The international governing body that might implement any changes is the United Nations (UN) Committee on the Peaceful Uses of Outer Space (COPUOS), which was first established in 1958 and which still promotes international cooperation with the treaties formed decades ago (di Pippo, 2014).

The technological environment has improved dramatically in the decades since humanity's first venture into orbit. The performance of space systems has improved since then and the mass driven down, in part due to increased miniaturization. A robust small satellite sector has emerged in the LEO economy that could be brought to cislunar space. Space systems have increased in reliability through a series of cutting-edge robotic exploration missions to Mars and the outer planets. Robotics and deployable systems continue to be developed and improved for use in deep space, especially for teleoperated or autonomous operations. However, there remain challenges, such as remaining concerns of radiation damage, challenges operating in the lunar night and in the abrasive lunar regolith, and lack of experience using a high degree of autonomy. Additionally, the launch vehicle sector presents challenges for any cislunar activities. Today's launch vehicles are currently very limited in their capability to deliver sizable payloads to cislunar space, and in general, launch vehicles are prone to delays, risk of failure, and high prices. Reviewing the status quo, both new and established global actors in the space industry are capable of advanced missions, but expansion out of Earth orbit will be constrained by significant remaining challenges.

1.1.3 Barriers

New ventures in cislunar space are likely to encounter barriers at the program, industry, and global levels. Some challenges were already mentioned above, which may prove to be barriers to program success. At the program level—whether the venture will be managed by a government agency or private company—common barriers for new space ventures include funding instability, developmental delays, delays or high costs acquiring licenses or other regulatory approval, delays and high costs acquiring a launch vehicle, and competition from other ventures. For cislunar space, example developmental delays might occur while filling capability gaps in autonomous operations or radiation protection for long-duration missions. Additionally, it might be quite challenging to find customers that makes a cislunar venture profitable, since the future existence of a market base in cislunar space other than specific government contracts is still uncertain. However, what might prove to be the biggest barrier to program success is the fact that complex systems will be operating in a hostile environment, where in the event of failure there are no teams that can simply go out to service the systems or receive spare parts in a matter of hours. The fact that most of the systems will be unproven in the cislunar environment exaggerates the risk. The risks and uncertainties associated with deep space missions are high, and high risks will drive development schedules and funding priorities, making many of the barriers interconnected. If any of the systems are to be crewed by human astronauts, the risks are only amplified.

Industry-level barriers might involve uncertainty in the launch services market. Uncertainty could exist in whether a launch vehicle will be available, especially if new vehicles must be developed to provide enough capability to deliver the systems out to cislunar space. Launch vehicles are historically plagued by delays to development and certification. Additionally, NASA has only begun to establish CLPS contracts in the United States, but there will likely be challenges to overcome as the proposed partners hammer out contracts, competition, and capability requirements,

let alone the uncertainties with whether similar contracts will be sustainable for other government or commercial customers.

Finally, global barriers might exist with regards to regulations or cooperation. For example, the UN COPUOS currently does not enforce regulations on the use of ISRU, but if that were to change, the regulations might restrict cislunar activities. Barriers might also exist if international partners are planned to assist in a venture but political priorities change that make the partnerships impossible.

1.2 Opportunities in On-Orbit Servicing, Assembly, and Manufacturing

Discussions about the role of on-orbit servicing, assembly, and manufacturing activities in the future of the space industry have occurred frequently, where sometimes these activities define a new sector and sometimes they exist as advanced capabilities in support of a sector. New opportunities are possible employing OSAM in both cislunar space and Earth orbit. The following sections will continue the SoSE Problem Definition phase and add to the discussion of the operational context, status quo, and barriers for new ventures as related to OSAM.

1.2.1 Operational Context

OSAM capabilities have been discussed for decades, dating back to astronaut servicing missions for Mir, Skylab, and the Hubble Space Telescope. Since then, there have been various proposals, demonstration missions, and federal agency activities to explore the value of advancing OSAM capabilities, such as:

- The Defense Advanced Research Projects Agency (DARPA) describes their Robotic Servicing of Geosynchronous Satellites (RSGS) program as "an idea whose time has come". RSGS, which kicked off in 2016 and which found a new commercial partner in March 2020, is intended to "improve satellite resilience

and create significiant opportunities for U.S. government and commercial satellite partners" (Roesler, 2016), (Erwin, 2020).

- Restore-L is a robotic spacecraft designed to "rendezvous with, grasp, refuel, and relocate a government-owned satellite to extend its life," currently under development at the Satellite Servicing Projects Division NASA Goddard Space Flight Center with industry partner Space Systems Lorral (NASA, 2016a). This Technology Demonstration Mission (TDM) is expected to launch in 2023.

- Space Infrastructure Dexterous Robot (SPIDER) is a secondary payload attached to Restore-L to include a 16-foot robotic arm and to demonstrate both assembly of a communications antenna and manufacturing of a 32-foot composite beam (NASA, 2016a).

- The Space Science and Technology Partnership Forum (with USAF, NASA, NRO, DARPA, and U.S. NRL) was established in 2015 to facilitate partner dialogue and create recommendations for investing in in-space assembly capabilities (Williams et al., 2018).

- Northrop Grumman's subsidiary, SpaceLogistics LLC, just completed its first successful docking of a Mission Extension Vehicle (MEV-1) with an Intelsat client satellite in February 2020, demonstrating their capability to extend the life of satellites through on-orbit servicing (Grumman, 2020).

- NASA's CubeSat Proximity Operations Demonstration (CPOD) is expected to launch in 2020, with the goal of demonstrating a unique docking device, imaging sensors, and small propulsion system for autonomous relative station-keeping and docking (NASA, 2016b).

- NASA funded the CIRAS (Commercial Infrastructure for Robotic Assembly and Services) project in collaboration with industry partners Northrop Grumman, their subsidiary SpaceLogistics, LLC, and the U.S. Naval Research Laboratory.

By summer 2018, the CIRAS project had developed and matured technologies for robotic assembly of large space structures (NASA, 2018).

- Made in Space is leading the charge for in-space manufacturing. They sent a 3D printer to the ISS in 2014 (Made In Space, 2019a), were awarded a contract in July 2019 to send "the first commercially-developed plastic recycling facility" to the ISS (Made In Space, 2019a), and were awarded a contract in October 2019 to send Archinaut, "the company's autonomous robotic manufacturing and assembly platform," to the ISS (Made In Space, 2019b).

Overall, OSAM has become a broad term with many proposed approaches and research areas. Entering 2020, there have even been enough demonstrations and development that make a commercial OSAM market feel as if its on the brink of becoming a reality. To narrow the scope, this thesis will look at strategies for a "persistent platform". A persistent platform is one approach where a system is created to serve as long-duration infrastructure to which other systems can dock or instruments can be plugged in. The objectives associated with a persistent platform are generally to enable improved maintainability, evolvability, or flexibility for in-space assets; to improve persistence and resilience of the space systems; and to advance science and technology progress (Williams et al., 2018), (NASA, 2016a). Such a platform can host different instru-

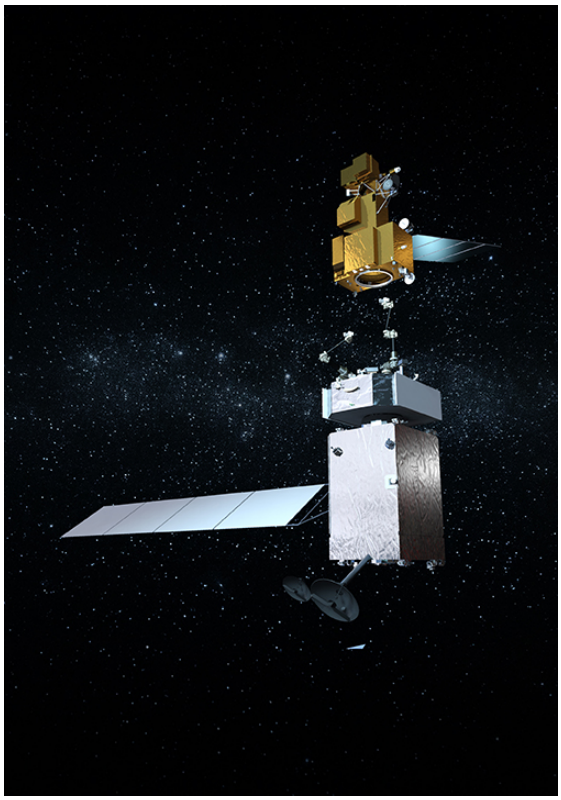


Figure 1.1: NASA's Restore-L (NASA, 2016a).

ments, payloads, or technology demonstrations by providing the required infrastructure in orbit. Customers can be supported from government, academia, and industry in order to make a persistent platform venture profitable for many years. Further, the ability to include any combination of servicing, assembly, or manufacturing capabilities is likely to enhance the platform's ability to fulfill those objectives. There are many possible strategies for designing a persistent platform, such as those suggested by the NASA Jet Propulsion Laboratory (Jet Propulsion Laboratory, 2018), Tethers Unlimited (Tethers Unlimited, 2017), the Aerospace Corporation (Aerospace Corporation, 2018), Space Systems Loral (Schwarz et al., 2018), or even the NASA Gateway (NASA, 2019c). The selected strategy must therefore be based on careful analysis of the stakeholders' preferences and the state of the technical, economic, and regulatory environments.

1.2.2 Status Quo

The technological environment for OSAM is defined by the presence of a small group of public and private actors that have been advancing the state of OSAM capabilities, both by maturing technologies and by organizing across government agencies to collaborate on funding priorities. This group includes DARPA, the Exploration and In-Space Services (ExIS) projects division (formerly known as the Satellite Servicing Projects Division) at NASA Goddard Space Flight Center, the Science and Technology Partnership Forum for in-space assembly (with the U.S. Air Force, NASA, National Reconnaissance Office, DARPA, and U.S. Naval Research Laboratory), Northrop Grumman, Space Systems Lorral/Maxar Technologies, and others. To add to this, the International Space Station (ISS) has given the space community twenty years of experience in design, operations, maintenance, supply management, and international partnerships. Drawing on the ISS experience, space systems hardware for operations in LEO have advanced to the point where long-duration platforms can support many different users over decades, given the proper care and maintenance.

Assembly capabilities have also developed through docking and berthing operations with the ISS and various autonomous satellite missions, such as the Autonomous Extravehicular Robotic Camera Sprint Vehicle in 1997, Japanese ETS-VII in 1998, U.S. Air Force Research Laboratory Experimental Small Satellite-10 (XSS-10) in 2003, and DARPA Orbital Express in 2007, among others (Spencer, 2015). Autonomous rendezvous, proximity operations, and docking are therefore sufficiently developed to enable in-space assembly. Although in-space servicing remains human-in-the-loop for now, there are companies that have been developing robotic servicing capabilities, mainly focused on robotic arms, grasping methods, and refueling.

Despite the recent advancement of many technologies that would enable OSAM, the persistent platform approach is inherently multi-faceted with unique technological, legal, and economic considerations. To add to the discussion of the state of the regulatory environment for space activities from the previous section, certain questions arise with regards to servicing, manufacturing, and liability. For example, there is uncertainty over who is liable if there is damage done while servicing. According to the Outer Space Treaty of 1967, Article IX, "harmful interference" is supposed to be avoided, but the Treaty merely suggests that a State worried about this "may request consultation" rather than establishing any more strict means of enforcement or punishment (United Nations Office for Outer Space Affairs, nd). Article VII establishes that the State that launches or procures the launch of an object and the State from which an object is launched are liable for damage. However, the playing field has grown significantly since the Treaty was written in the 1960s to include a wide variety of non-governmental entities. Today, the industry might need international regulatory changes to establish protocol for permission to perform servicing and protocol if accidental damage is done to assets in space while servicing.

Additionally, international regulatory changes might be needed with regards to ownership of assets in space. In Article VIII of the Treaty, "a State... on whose registry an object launched into space is carried shall retain jurisdiction and control over such object... including objects landed or constructed on a celestial body"

(United Nations Office for Outer Space Affairs, nd). However, this may introduce complications for new OSAM ventures. OSAM strategies discussed by industry today can involve swapping components, selling manufactured items to other actors, recycling dead spacecraft left in graveyard orbits, and operating payloads on permanent platforms. In these cases, the industry might advocate for "jurisdiction and control" to be passed instead to the owner of the servicing/recycling spacecraft, or to the operator of the platform rather than the payload owner. Finally, a State which oversees non-governmental entities operating in space does have authorization and supervisory powers, per the Treaty, and additionally has the power to enact taxes on services rendered in space.

These are just a few examples of foreseen changes that may be needed for a future OSAM market. Reasonable precedents can be considered to have been set by the satellite communications market, where commercial ventures located in one country already provide communications services to customers around the globe. The new categories of servicing for in-space repair and in-space manufacturing may also fall under existing regulations and taxation relevant to the space-based communications market. However, the global community via the UN COPUOS is not yet able to enforce legally-binding regulations on possible bad actors in space, and on-orbit failures are currently only able to be mitigated with spacecraft insurance plans.

1.2.3 Barriers

New space ventures which include OSAM will experience many of the same program-level barriers mentioned already in cislunar space. However, staying in LEO would bypass at least some of the barriers, such as those associated with acquiring a capable launch vehicle and being days of travel away from the operations in case resupply or assistance of any sort is needed. Industry-level barriers might still involve uncertainty in the launch services market; however, this is more likely to be a barrier because any significant improvement in launch vehicle availability and/or pricing might actually

weaken the business case for OSAM. The potential to profit from servicing expensive, in-space assets is likely to worsen if the price to launch a new replacement drops. Finally, additional barriers could emerge with respect to the regulations and liability uncertainty surrounding on-orbit servicing and manufacturing.

1.3 Research Objective

There are interesting opportunities in a future cislunar economy and in a future OSAM market, but these opportunities clearly also have distinct challenges and barriers. To explore the opportunities, this thesis will focus on the following two research questions:

- How can various existing tools for value analysis of space systems or SoS be integrated to improve decision support and to improve usability?
- Which design features for a persistent OSAM platform capable of hosting customer payloads can be varied to best balance value (e.g., flexibility, complexity) and life cycle cost?

This thesis will employ an SoSE modeling and analysis process to evaluate one OSAM concept—a persistent platform with OSAM. Already in this chapter, SoS problem definition has been started by describing the operational context, status quo, and barriers associated with the opportunity. Next, this thesis will provide a review of SoSE modeling and analysis methods, Value-Centric Analysis (VCA), and architecture building tools in Chapter Two. In Chapter Three, the development of the Modeling Architectures and Parameterization of Spacecraft (MAPS) environment will be presented for implementing these methods. In Chapter Four, the methods will be demonstrated for a case study on a persistent platform suitable for an emerging OSAM market. Finally, Chapter Five will conclude with a summary and discussion of potential future work.

CHAPTER 2. REVIEW OF SOSE TOOLS AND METHODS

This chapter will present a brief review of three topics: System-of-Systems Engineering, Value-Centric Analysis, and use of architecture building tools. This thesis will implement methods from these topics in order to respond to the research questions; therefore, particular focus will be given to implementation for space systems problems.

2.1 System-of-Systems Engineering

The opportunities that this thesis wishes to addresses, whether an opportunity in cislunar space or in an emerging OSAM market, clearly must account for the many technological, operational, political, and economic factors that are involved in the complex web of the space industry. These many factors were identified in Chapter One. SoSE is a discipline that is aptly suited to characterize such opportunities, so the current state of SoSE will be reviewed. Then, appropriate modeling and analysis methods within the context of SoSE can be applied to an example opportunity.

2.1.1 SoSE Status Quo

The discipline of SoSE was born as a result of a group of researchers identifying that the characteristics of SoS that set them apart from traditional systems or even highly complex systems (DeLaurentis, 2005). An SoS is special in a way that it reflects some attributes of a "system" and has one or more key objectives, but it also has components that are "systems" in their own right, with their own independent objectives. Maier's widely accepted description includes five distinguishing characteristics (Maier, 1998):

1. Operational independence
2. Managerial independence
3. Geographical distribution
4. Evolutionary development
5. Emergent behavior

Component systems in an SoS often exhibit operational and managerial independence, and they are often agents in multiple domains. SoSE is therefore used for shaping an appropriate frame of reference for multi-domain problems, for developing multi-agent modeling and simulation, and for enabling architecture-level decision support. SoSE was developed specifically to address the need for novel methods where traditional systems engineering is not suitable. Some questions that these methods strive to answer include (DeLaurentis et al., 2019):

- Which systems/functions/resources should make up the composition of the SoS?
- How do operational and developmental interdependencies shape the topology, and how would failures impact the architecture?
- How do different control strategies change the SoS behavior? What is the ideal level of independence, and how are systems incentivized to contribute to the SoS-level objective(s)?
- How do different stakeholder objectives and preferences impact the SoS value, cost, or risk?

These questions introduce different classes of design variables: composition, topology, and control (DeLaurentis et al., 2019). It is critical that any SoS modeling and analysis effort properly represents the variables during concept analysis. DeLaurentis (2005) first introduced the "DAI process" for SoS modeling and analysis, involving

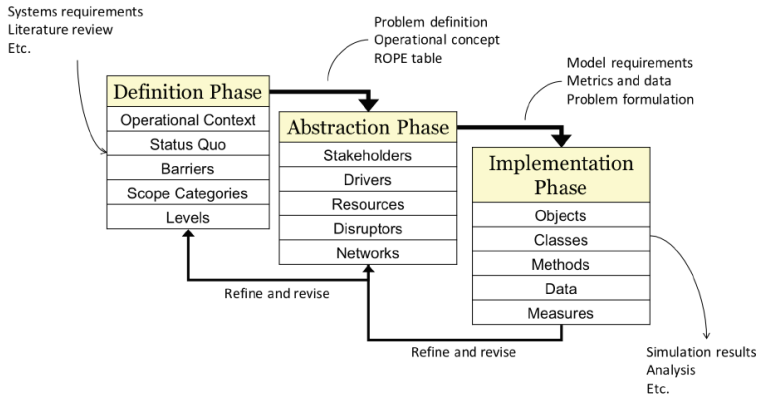


Figure 2.1: Three phases for SoS modeling and analysis (DeLaurentis et al., 2019)

three phases: Definition, Abstraction, and Implementation. The main components of these phases are depicted in Fig. 2.1. The phases are described as follows (DeLaurentis, 2005):

1. **Definition Phase:** To understand the problem, including description of the SoS operational context, status quo, and barriers, plus definition of the taxonomy.

2. **Abstraction Phase:** To frame the problem, including identification of the stakeholders, resources, and drivers, plus definition of the SoS as a network with interactions and interrelationships, if desired.

3. **Implementation Phase:** To analyze the problem, including creation and application of the model or simulation, interpreting results, and exploring alternatives.

The DAI process guides the case study for this thesis, beginning with many components of the Problem Definition phase provided in Chapter One. In particular for the case study, the process is employed to answer the first and last questions in the bulleted list above.

2.1.2 Taxonomy

An agreed-upon taxonomy is important for clear understanding and implementation of the DAI process, so a specific taxonomy appropriate for space systems has been defined and is used throughout this thesis. Key terms are defined below.

- A *Stakeholder* is defined as a human agent who has some control, whether implicit or explicit, of the system design or operation. Stakeholders may represent decision-makers at many levels, including government actors, the firm, the customer, the engineer, or the user (Ross et al., 2004), (Parnell, 2016).

- A *Decision Opportunity*, though can be seen as an opportunity or a problem, is an activity where decision-makers may spend resources to achieve defined objectives (Parnell, 2016).

- An *Objective* is a defined goal or desire for stakeholders that can be traced to the functions in a functional architecture (Parnell, 2016).

- A *Value Measure* is a means for a stakeholder to quantify how well an objective is achieved (Parnell, 2016).

- A *Scenario* is a distinct series of operations that may be a variation of the mission. The operations are specifically defined to allow value measures to be assessed. (Parnell, 2016).

- The *Tradespace* is "a multidimensional space that defines the context for the decision, defines bounds to the region of interest, and enables Pareto optimal solutions for complex, multiple-stakeholder decisions" (Parnell, 2016).

- A *Concept* is a preliminary description of a means to perform functions to achieve objectives with a set of resources and operations. A concept is defined prior to defining a specific architecture or design (Parnell, 2016), (Friedenthal et al., 2019).

- A *Functional Architecture* is a model describing a set of functions for a concept, typically as a sequence of events towards achieving objectives (Raz et al., 2018).
- A *Physical Architecture* is a model describing the physical systems of a concept at the highest level, including the composition of physical systems (i.e., modules and assemblies), and system interactions (Raz et al., 2018), (Parnell, 2016), (DeLaurentis et al., 2019).
- A *Design* is a description of specific systems, system properties, and interfaces within an architecture (Parnell, 2016), (Richardson et al., 2010).
- A *Component* is the lowest level of descriptor for a physical system design, which may be commercial-off-the-shelf parts representing zero development effort to the designer. Examples include a solar cell or a reaction wheel.
- A *Subsystem* is a class defined by a collection of components reflecting the most common function allocation for space systems. There are seven common functional subsystems, and the required ones for each spacecraft are avionics, power, and communications.
- A *Module* is a class defined by a free-flying collection of linked subsystems. These links may be structural or represent the flow of electrical power, data, thermal heat, or fluid. Modules are usually assembled on the ground, usually with the intention that the links are permanent (except in the case of on-orbit servicing or assembly).
- An *Assembly* is a class defined by the temporary linking of multiple modules. These links may be structural or represent the flow of electrical power, data, thermal heat, or fluid. At any given time or location during the simulation, an assembly might form or change through docking, berthing, grappling, etc.

2.2 Value-Centric Analysis to Evaluate Space Systems Architectures

Value-Centric Analysis (VCA) stems from Value-Focused Thinking (VFT) and is a method that can be used for SoSE analysis. This section will explore the foundation and application of VCA.

2.2.1 Background of VCA

If a decision opportunity is defined to answer the SoS analysis question, "Which systems/functions/resources should make up the composition of the SoS?", then there must clearly be a way to compare the composition of architecture alternatives. Fundamentally, systems are designed to perform functions that achieve stakeholder objectives and provide value. The levels of achievement of stakeholder objectives, or "value measures", should therefore be used as a means to compare architectures. This is the foundation of VCA. VCA, as defined by (Richardson et al., 2010), "uses a system value model to measure the relative attractiveness of alternative architectures, materiel solutions, and conceptual system designs." In the case that there are stakeholders with competing objectives, sacrifices—or trade-offs— may be required of the value measures (Parnell, 2016). In this case, VCA becomes a particular method of trade-off analysis. Trade-off analysis is now a well-developed practice for systems engineers to properly characterize the design space, evaluate alternatives, trade measures based on different stakeholder preferences, and inform decision-making.

Further, from a design standpoint, VCA can enable consideration of multiple phases of the design cycle already in the initial conceptual phase. Value measures can be included that reflect "manufacturability and assembly, deployment, operations, maintenance, and decommission" (Ross et al., 2004). Architecture-level measures can be designed to consider these later phases. When a design team performs thorough trade-off analysis with diverse value measures, they can identify common characteristics or technologies that provide high value and warrant further investigation. Therefore, architecture alternatives can be down-selected for further design

effort, and stakeholders can make decisions about future investment in the common characteristics or technologies.

VCA has been selected in comparison to cost-centric analysis, where an architecture is selected based on its satisfaction of system performance requirements at minimum cost (Brown and Eremenko, 2008). Instead, the decision opportunity can be solved by explicitly designing to maximize value for stakeholders while treating cost as a wholly separate trade.

Value-Focused Thinking

Value-Focused Thinking (VFT) is the philosophical approach introduced by Ralph Keeney (1996), from which VCA stems. In this seminal work, Keeney describes the the many benefits of the approach. These benefits include, for example, VFT's ability to support strategic planning of an organization, engage decision makers, improve communication between stakeholders, identify and evaluate decision opportunities, and ultimately make better decisions (Parnell, 2016). To successfully realize these benefits, the four key ideas of VFT are as follows (Keeney, 1996):

1. Start first with values
2. Generate better alternatives
3. Create decision opportunities
4. Use values to evaluate alternatives

These ideas form the basis for the method outlined by VCA within the decision management process, as will be laid out in the steps described in the next section.

Comparison to Other Methods

As mentioned, VCA has been selected in comparison to a cost-centric analysis of architecture alternatives. Additionally, it is recognized that there are other value-

based methods discussed in the literature other than VCA. A brief overview is provided for understanding of the status quo of architecture analysis.

Utility theory was expanded by Keeney and Raiffa to become Multi-Attribute Utility Analysis (MAUA) (Keeney et al., 1993). Later, researchers at the Massachussetts Institute of Technology (MIT) developed the Generalized Information Network Analysis (GINA) to convert space systems into an information network diagram to help identify a design variable vector and apply optimization methods. Multi-Attribute Tradespace Exploration (MATE)—and when combined with concurrent engineering, MATE-CON—was GINA's successor, developed at MIT by (Ross, 2003). In MATE, attributes are identified to reflect stakeholder objectives and are evaluated independently, much the same as the value measures of VCA. Later studies stemming from the DARPA System F6 program, which kicked off in 2008, guides tradespace exploration using an architecture-level value model and discusses the role of value metrics in systems engineering. This seems to mark the emergence and popularization of the term "value-centric design" (Brown and Eremenko, 2008).

Value-Centric Design (VCD) is a method that also distinguishes alternatives based on a value model; however, VCD and VCA answer different questions and occur at different phases of the design process (Richardson et al., 2010). Beginning typically after a Preliminary Design Review, VCD is applied to lower level system design and must use a frozen concept and frozen value model, since many subsystems will be designed in parallel. This is in comparison to VCA, which begins the design process, where many concepts can be evaluated and where the value model is expected to evolve (Richardson et al., 2010). Researchers from DARPA made progress to couple these two methods and named it Value-Centric Design Methodology (VCDM) (Richardson et al., 2010); however, it still appears that the distinction between the two was maintained. Later, Value-Driven Design (VDD) was named by members of collaborating AIAA Technical Committees, but VDD has also been named as the precursor to VCDM, and therefore is likely equivalent to VCD (Collopy et al., 2012).

2.2.2 VCA in the Decision Management Process

The primary goal of concept analysis is to support decision-making. (Ross et al., 2004) and (Williams et al., 2018) emphasize the importance of capturing stakeholder values and preferences early in the decision process, so they can make decisions based on transparent trades. (Parnell, 2016) adds that stakeholders are more likely to be willing to participate in the process "when [they] see that their objectives are included". Therefore, explicit analysis of value measures in VCA improves the communication amongst decision-makers and design teams and improves the evaluation of alternatives.

An understanding of the steps towards effective decision management is critical when beginning a trade study. The International Council on Systems Engineering (INCOSE), IEEE Computer Society, and Stevens Institute of Technology have together published the Systems Engineering Body of Knowledge (SEBoK) compendium. The decision management process in SEBoK (Madachy et al., 2019) is designed to "provide a structured, analytical framework for objectively identifying, characterizing, and evaluating a set of alternatives for a decision at any point in the life cycle and [to] select the most beneficial course of action" (Parnell, 2016). This process is outlined in Fig. 2.2.

This thesis hopes to contribute by (1) improving a user's ability to apply the decision management process for space systems concepts and (2) integrating tools to evaluate non-traditional value measures for space systems. Though no part of the SEBoK process should be neglected, this thesis narrows the focus to the following steps while creating an initial environment for concept evaluation:

1. Develop objectives and measures
2. Generate creative alternatives
3. Assess alternatives via deterministic analysis
4. Synthesize results

5. Communicate trade-offs

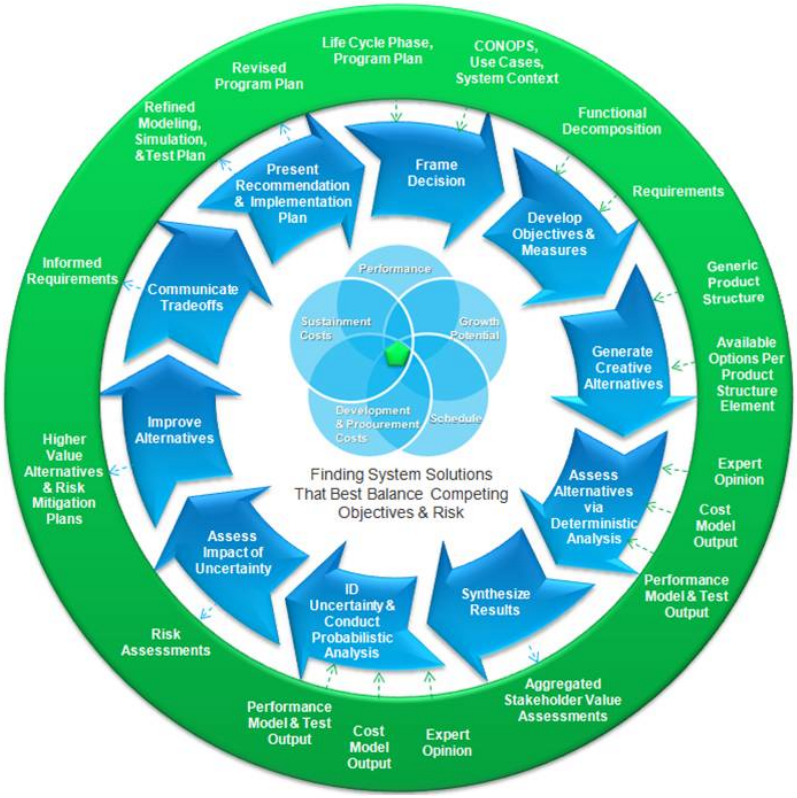


Figure 2.2: The decision management process as presented by the SEBoK (Madachy et al., 2019)

First, analysts identify the stakeholders' objective(s). Objectives are matched to functions and further to value measures through the Functional Value Hierarchy process. A literature review of various value measures is presented in the following sections. The next step in the decision management process is to generate creative alternatives, which can be accomplished several ways. Brainstorming and creativity activities are generally good means of producing alternatives for mission concepts and technologies (Parnell, 2016), (Cross and Roy, 2000). Alternatives can also be enumerated by varying system parameters within user-defined bounds and constraints. Lastly, space systems architecture alternatives can be enumerated using numerical methods to distribute desired payloads/subsystems amongst a number of free-flying spacecraft. Each alternative is then assessed using the chosen value measures.

In the next step, synthesis of results occurs as the value measures data are incorporated into an architecture-level value model. With multiple value measures, it is necessary to follow methods for Multiple Objective Decision Analysis (MODA) (Parnell, 2016). Though tradeable value measures could be graphed for individual comparisons, MODA literature suggests it is often more beneficial to use an additive value function. For the additive value function as described by (Parnell, 2016):

1. The value measure data is normalized using single-dimensional value functions on an interval scale from 0-1, producing $v_i(x_{ji})$
2. Stakeholder-defined swing weights, w_i, for each value measure are defined on a ratio scale, where zero indicates no value. Also, Equation 2.1 must be satisfied.
3. Total value, $v(x_j)$, is computed as the weighted sum of the normalized values using Equation 2.2.

$$\sum_{i=1}^{n} w_i = 1 \tag{2.1}$$

$$v_j = \sum_{i=1}^{n} w_i v_i(x_{ij}) \tag{2.2}$$

where

i = index of the value measures, 1,...,n

j = index of the alternatives, 1,...,m

w_i = swing weight for ith value measure

x_{ji} = jth alternative's score for the ith value measure

$v_i(x_{ij})$ = normalized value of the score x_{ji}

v_j = total value for jth alternative

There are several important things to note about the additive value function. [1] First, cost is intentionally modeled separately such that total value and total cost

[1]Please see (Parnell, 2016) Chapter Two for more additional information on the mathematical foundation of the additive value function and requirements for its use.

can be independently traded. Also, this process assumes that all the value measures have different units, so they can't all be combined in a different way. Finally, the presence of multiple stakeholders or scenarios may mean different sets of priorities (swing weights) and different sets of analysis results.

The final steps of the decision-making process—of those that will be covered in this thesis—are to synthesize results and communicate trade-offs. Once a composite measure of the total value as well as total cost has been computed, various graphs can be created to represent the results of the trade study. Visualization of the results may be in the form of Pareto frontiers, bar charts, spider plots, or other graphs. Pareto frontiers are particularly useful to communicate how certain design characteristics impact the trade-off between value and cost.

2.2.3 Traditional Measures of Performance

Measures of performance used in trade studies are commonly physical attributes of the architecture. For space systems architectures, this might include total mass, communications bandwidth, or coverage. Mass is a performance measure that is sometimes used to measure improvements in the design or draw parallels to financial cost. However, with the intent usually being to minimize mass and minimize life cycle cost, it would be equally valid to refer to mass as a cost. Either way, it is important for the user to keep in mind during trade-off studies that the objective is to *minimize costs* and *maximize performance*. If mass is considered a performance measure, it would be common practice in optimization to use one minus the mass, $1 - M$, to actually minimize this measure for space systems.

There are several options for mass-based measures:

- IMLEO
- Launch mass
- Mass to surface

- Mass robustness penalty
- Mass + MRP

The selection of the most appropriate measure is dependent on the type of mission and stakeholder preference. For example, there exists the "fractionated" architecture, wherein spacecraft functional subsystems are distributed across multiple free-flying modules with wireless connections, in comparison to a "monolithic" architecture, which is the more traditional method of permanently constructing a single spacecraft with all the desired subsystems and payloads (Brown and Eremenko, 2008). The measure Mass Robustness Penalty (MRP) is introduced by (Neema et al., 2018) as a modifier for fractionated architectures, since fractionated architectures are expected to have a higher mass but also "are argued to provide better flexibility and robustness over [a] monolithic architecture". Considering the case where replacement spacecraft would need to be delivered in the case of any payload failure(s), MRP accounts for the mass that would have to be launched based on failure probabilities, as defined using Eq. 2.3 (Neema et al., 2018):

$$MRP = \sum_{m=1}^{M} \sum_{i=1}^{m} 1_i(j) M_i f_j \tag{2.3}$$

where

M = number of modules

m = number of fractionatable elements

M_m = mass of the mth module

f_j = failure probability of the jth fractionatable element

$1_i(j)$ = indicator function, defined in Eq. 2.4:

$$1_i(j) = \begin{cases} 1 & j \in i \\ 0 & else \end{cases} \tag{2.4}$$

It may be common practice for aerospace engineers to balance architecture design decisions based on the cost or mass penalty they may incur. However, there are

several other important value measures—such as flexibility, complexity, robustness, and others—that have been developed in academia and industry. These measures have the potential to be integrated into the systematic design process rather than be considered ad hoc or qualitatively, and many sources in literature have presented methodologies, tools, and frameworks for doing so (Ross et al., 2004), (Brown and Eremenko, 2008), (Mathieu and Weigel, 2005), (Neema et al., 2018), (Tamaskar et al., 2014).

2.2.4 Flexibility

Flexibility is a nontraditional value measure that is assessed at the architecture level and that identifies valuable alternatives based on considerations for the operations phase. Flexibility is especially important as a value measure because the world is inherently uncertain and imperfect. Flexibility gives a stakeholder (i.e. developer, owner, or operator) options to alter the system(s) in some way in the event of changes in requirements, technologies, or environment (Brown and Eremenko, 2008), (Neema et al., 2018). Such changes may be detrimental and require repairs, while others may provide opportunity for growth. (Brown and Eremenko, 2008) lists several possible changes, or "perturbations", that may occur during the life cycle of space systems, such as funding fluctuation, launch failure, or on-orbit failure.

In literature, flexibility is sometimes considered a category to encompass value measures such as adaptability, scalability, and maintainability, evolvability (Brown and Eremenko, 2008), (Mathieu and Weigel, 2005). Categorizing these value measures together is done to acknowledge the various types of changes and the types of responses which may occur during a system's lifetime (Mathieu and Weigel, 2005). Each of these value measures could be assessed using models for individual operations scenarios, cost, or other options. Four key value measures within this category are defined as follows (adapted from (Brown and Eremenko, 2008)):

- Adaptability: the ability to reconfigure a system to meet new needs or circumstances
- Scalability: the ability to add components, capability, or performance to a system throughout its lifetime
- Maintainability: the ability to replace components that have failed or are near end of life
- Evolvability: the ability to replace and upgrade components due to technology obsolescence

Other sources in literature define flexibility as a value measure itself. According to (Mathieu and Weigel, 2005), flexibility is "the ability of a system to be modified to do jobs not originally included in the requirements, namely a change in function". Separate from "maintainability" and "scalability", they assessed all three alongside each other using defined operational scenarios and using cost as the measure. The analysis in (Mathieu and Weigel, 2005) was performed on fractionated spacecraft architectures, including three missions (communications, navigation, and sensing). Their results showed that in the scenario where a replacement payload is required, increased fractionation level corresponded with more flexibility. The measure of flexibility in their analysis meant a significant decrease in both the financial cost and mass cost of the replacement payload module—e.g., as low as 30% of the mass of the comparable monolithic spacecraft architecture for the communications mission.

(Neema et al., 2018) also define flexibility as an individual value measure. Their measure is based on the quantity of design changes during the reconfiguring of a reference architecture design into other architecture designs, which need to meet some performance target/threshold. Inspired by network theory, the Neema flexibility model follows those presented by (Nilchiani and Hastings, 2005) and (Moses, 2012), "where a system is flexible if it is relatively easy to add new nodes and connect them to the existing nodes for adding new functions or to modify existing functions." Their

model was also presented with a proposed visualization technique of a polar plot, where the axes display the number of design changes and the normalized performance change from the design reconfiguring.

Although (Neema et al., 2018) did not discuss it, their flexibility measure could be applied for a variety of scenarios similar to those from (Mathieu and Weigel, 2005), which represent possible changes not just to performance but also requirements, technologies, or environment. The application to specific scenarios would be an improvement to the model, since the scenarios would add specific meaning by which flexibility would be understood. For instance, in a scenario for "adaptability," the stakeholder might change a requirement to define a new performance target for the system. The Neema model can be used virtually in the same way as the original application; an architecture is evaluated for how flexible, or adaptable, it is to be reconfigured into designs that meet the new performance target. However, this is an improvement because there is context for understanding the flexibility measure traced to a change in a requirement.

The four value measures within the category of flexibility will be integrated into the architecture analysis for this thesis, along with scenarios like the one discussed above. Analysis will include an adapted application of the Neema model for specific scenarios, as well as other adaptations to explore the substitution of cost and complexity in place of performance in the Neema model.

2.2.5 Complexity

Space systems in particular are often described as being highly complex, and it is deemed valuable to be able to trade this value measure (Holtta-Otto and de Weck, 2007), (Neema et al., 2018), (Ross et al., 2004). It is an observed trend in the space industry of the 21st Century that there is an increased number of components, increased coupling, and the presence of feedback loops in space systems. This in turn increases the design and systems engineering effort to develop the systems. Further,

systems may be developed with "additional redundancy to maintain the desired level of reliability in operation." These trends for already complex systems may increase costs (both financial and mass costs, for space systems) and increase risk of developmental delays. Therefore, complexity management is a valuable effort during design (Tamaskar, 2014).

(Tamaskar, 2014) previously performed a review of complexity measures in literature. The focus was on specifically design and development complexity for aerospace systems. Network theoretic-based measures have been presented in several works, which collectively base the measure on coupling, correlation, dependency, size, and/or hierarchy. (Tamaskar et al., 2014) developed a network theoretic-based measure too and proposed that the most important aspects of complexity include coupling, directionality, and feedback loops. In addition, their model accounts for dependency by adjusting the network link weights and accounts for modularity by introducing an integration coefficient. This model was applied to an example spacecraft, and conclusions were drawn about the system with and without modularity, about the differences in coupling complexity across subsystems, and about the significance of integration complexity. The results are displayed in graphs of complexity vs. cost and vs. performance (Tamaskar, 2014).

(Neema et al., 2018) also applies Tamaskar's complexity metric to space systems, this time with a focus on fractionated architectures. In addition to the complexity a single spacecraft module, Neema et al. compute the complexity of the full fractionated architecture by summing module complexity values. This is justified by the expected correlation between an increase in the number of modules and an increase in complexity (due to increased number of components and direct and indirect coupling). In the results, it is observed that a higher quantity of modules does not guarantee increased architecture complexity, since complexity depends on the characteristics of the payloads in the modules. However, the average architecture complexity (for architectures with the same quantity of modules) does increase with the quantity of modules.

(Raz and DeLaurentis, 2017) also use Tamaskar's model, but they propose a modification to account not just for the interactions of systems but also how complicated each system is. Their coupling complexity measure is modified such that the weights in each summation are multiplied by a system complicatedness measure (SCM). The SCM for a system is a sum of f_{c-il}, the function complicated-ness of the lth function of the ith system. They admit, however, that identifying that function complicatedness "remains subjective and requires domain expertise."

(Pugliese and Nilchiani, 2017) explore three options for metrics to compute structural complexity. Similar to Tamaskar's model, they proposed that a complexity measure can be based on the structural graph of a system. However, their measure is based on graph energy, with 3 options for the graph: (1) adjacency matrix, (2) Laplacian matrix, (3) any matrix, as originally proposed by (Cavers et al., 2010) and applied to a normalized Laplacian matrix. The basis on graph energy also appears in other sources, such as (Sinha and de Weck, 2013). Additionally, the research of Pugliese and Nilchiani focused on fractionated architectures, where satellites could only be of one of three fraction types, largely based on the presence of payload(s) and the ability to communicate with external systems. They concluded that all three metrics resulted in the opposite correlation than expected between number of modules (or density) and the proposed complexity measure. What is commonly expected is that complexity generally increases with the quantity of modules.

This thesis will integrate the Tamaskar complexity model for architecture-level space systems analysis. The modification proposed by Raz and DeLaurentis will not be used, though the modification is identified as an area for potential future work, should the function complicatedness variable be considered less subjective in the future.

2.2.6 Life Cycle Cost

Finally, an architecture must be assessed for its financial cost. The life cycle cost for space systems include: design, development, testing, manufacturing, launch, and operations. Several cost models or cost estimating relationships (CERs) exist, some of which are publicly available and some proprietary. Examples of publicly available models created for spacecraft are the Advanced Mission Cost Model (AMCM), the Unmanned Systems Cost Model (USCM), and the NASA Instrument Cost Model (NICM). As can be inferred from their names, these models should be used in different scenarios and each have their own strengths.

The AMCM is one model that was designed to provide an estimate of design, development, test, and engineering (DDT&E) and production cost that is not just mass-based but also dependent on various parameters (Larson and Wertz, 1992). The AMCM parameters include: estimated difficulty, block/design generation, expected initial year of operation, a specification value for the type of mission (e.g. human habitat or planetary lander) and quantity of units. Each of these parameters can be assessed at any level of abstraction, providing the capability of creating detailed cost estimates at the component level. The AMCM has primarily been used for human mission cost analysis (Jones, 2015), (Jones, 2018). For future commercial ventures, including the quantity of units in the cost model may be quite valuable, where cost reductions from multiple copies of components may relied upon for affordability and profitability. However, the AMCM was not designed specifically for today's small spacecraft nor the COTS trend.

The USCM is a cost model created by the U.S. Air Force Space and Missile Systems Center for unmanned space missions and is described in *Space Mission Engineering: The New SMAD* (Wertz et al., 2011). The data from 44 military, NASA, and commercial satellites had been compiled in a database to create version 8 of the USCM as of the writing of *Space Mission Engineering* in 2011; however, as of 2020, a non-proprietary version 10 had been released with over 100 satellites in the

database and a more regular update frequency, according to USCM Online (US Air Force, Space and Missile Systems Center, nd). Compared to the AMCM, the USCM provides a set of both non-recurring and recurring CERs for individual subsystems and other categories based on several parameters, depending on the category. These parameters include the subsystem mass, number of communications channels, and propellant tank volume, among others (Wertz et al., 2011). The USCM only estimates payload costs for a communications payload, though, so a different model should be used for any other type of payload.

The NICM was created by the NASA Jet Propulsion Laboratory to estimate payload instrument costs, and version IIIC was released in 2010 (Wertz et al., 2011). Five instrument categories are listed, such as Optical Earth-Observing, Microwave, or Particles, so each payload in a given spacecraft can be assessed using a different category. The parameters used in this model vary depending on the instrument category, including instrument mass, instrument power, Technology Readiness Level (TRL), design life, and others. The NICM can be used together with the USCM to provide a complete picture of the development and production of space systems.

Beyond DDT&E and production costs, cost estimates should be made for launch and operations costs. Launch costs can be estimated simply by using publicly available launch price data. This price data could be used either for the price of a full launch vehicle or the price per kilogram for secondary payloads, depending on the payload size and mass and the payload-to-orbit capacity of the vehicle. For example, SpaceX Falcon 9 and Falcon Heavy launch vehicles have publicly available price and performance data provided on the company's website.

When using the USCM, there are CERs for recurring costs such as use of the aerospace ground equipment, launch operations and orbital support. When using the AMCM, there is no CER for annual operations costs. In this case, the annual operations cost can be estimated simply as 10% of the DDT&E and production cost from the AMCM (Jones, 2018). Of course, this is a low-fidelity estimate, and new paradigms in the space industry may impact this category of cost, too.

As in all models, certain limitations exist for the three cost models discussed here. Various new paradigms may impact future costs that are not reflected in historical data, just as small satellites have been changing the game through miniaturization and increased COTS availability over the last decade. However, many discussed or expected industry improvements or cost savings may not have been realized yet. For example, it is suggested that fractionated spacecraft could yield cost savings from increased autonomy during in-orbit operations; however, some argue that "conventional wisdom" suggests this type of architecture would actually be "a more complex and costly undertaking" with multiple interacting spacecraft rather than individual spacecraft. (Brown and Eremenko, 2008). Higher fidelity cost estimates require much more information about the time frame, expectations, and operators. The three models presented here, however, should be acceptable for architecture comparison, especially at the conceptual phase of design.

2.3 Architecture Modeling and Analysis Toolsuites

2.3.1 Industry Toolsuites: DARPA F6 Program

Several toolsuites have been created in the aerospace industry for rapid assessment of architectures. DARPA began the System F6 Phase 1 in 2008 to fund conceptual design studies by Boeing (McCormick et al., 2009), Lockheed Martin (Maciuca et al., 2010), Northrop Grumman, and Orbital Sciences (Eichenberg-Bicknell et al., 2009), as reviewed by (Brown and Eremenko, 2008). The purpose of the "Future Fast, Flexible, Fractionated, Free-Flying Spacecraft united by Information eXchange" (F6) Program was to explore and demonstrate a fractionated spacecraft architecture, investigating whether fractionated spacecraft could be "more flexible, robust, responsive, and ultimately more cost-effective space systems" compared to traditional monolithic spacecraft. DARPA provided the teams with performance objectives but very few specific mission requirements, and the teams applied VCDM to evaluate net present value for architecture options, similar to that presented in (Brown and Eremenko,

2008). Many of the industry partners created effective architecture-building frameworks and tools to perform such analysis.

In the industry partners' frameworks, system models included cost models, benefit models, risk evaluation, automated design generators, and/or optimizers. Boeing's RAFTIMATE generated various design configurations (McCormick et al., 2009). Lockheed Martin used MIT's GINA tool to generate configurations, and MIT Space Lab's Time-Expanded Decision Network optimization framework (Maciuca et al., 2010). Orbital Sciences used the Georgia Institute of Technology (GeorgiaTech) GT-FAST automated design tool, and developed the PIVOT tool for optimization (Eichenberg-Bicknell et al., 2009).

2.3.2 Government Toolsuite: EXAMINE

NASA also has a handful of separate tools built to perform mission, architecture, and campaign analyses. Created at NASA Langley Research Center in 2008, the Exploration Architecture Model for IN-space and Earth-to-orbit (EXAMINE) framework was built to enable: "1) a significantly larger fraction of an architecture trade space to be assessed in a given study time frame; and 2) the complex element-to-element and element-to-system relationships to be quantitatively explored earlier in the design process" (Komar et al., 2008). EXAMINE is an excel-based tool that has been demonstrated on various human exploration problems such as a Mars transportation system architecture and a Constellation propellant options study. EXAMINE includes the following capabilities:

- "Sensitivity analysis and Pareto graphing of architecture attributes
- Sensitivity analysis for mission requirements
- Assessment of gear ratios for system masses
- Verification and tracking of performance metrics
- Assessment of performance reserves

- Tradespace assessment of architecture and concept alternatives" (Komar et al., 2008)

These capabilities were provided by combining multiple tools in different Excel worksheets. Together, they made EXAMINE valuable with the ability to run quick trade studies at the conceptual level to explore the mission architecture design tradespace, albeit with some simplified assumptions. As with most architecture analysis of alternatives, it is noted that numbers out of the tool were not considered useful as precise designs so much as they were useful for comparison between designs. EXAMINE is still used today by NASA to effectively investigate concept feasibility.

2.3.3 Academic Toolsuites: MATE-CON, GT-FAST, and AWB

Already mentioned as part of the toolsuites in use by industry, academic toolsuites have played a significant role in the architecture analysis field. MIT had developed Multi-Attribute Tradespace Exploration with concurrent design (MATE-CON) by 2004, forming it into a comprehensive framework that addresses engagement of multiple stakeholders, the creation of multi-attribute utility functions through interviews, concept design space exploration, and more detailed concurrent design (Ross et al., 2004). MATE-CON's role in advancing the status quo for VCA has already been presented. The process involves:

1. Need identification to understand problem scope
2. User interviews to develop single- and multi-attribute utility functions
3. Tradespace formation, which is done by determining the design variables, with elimination of variables that would have a weak impact on the design
4. Architecture enumeration using their modular software

5. More detailed design using CalTech's ICEMaker software, allowing communication for concurrent design, and with the oversight of a MATE-CON chair to continuously evaluate utility and cost

MATE-CON is certainly a comprehensive and thorough framework for applying VCA principles. MATE-CON doesn't, however, introduce more non-traditional architecture value measures—e.g., flexibility and complexity—when defining the utility functions.

Further south, the GeorgiaTech F6 Architecture Synthesis Tool (GT-FAST) was developed in 2009 for the DARPA F6 program (Lafleur and Saleh, 2009b), (Lafleur and Saleh, 2009a). GT-FAST was developed to address remaining challenges in enumerating many possible alternatives for fractionated architectures and in performing architecture-level value and cost analysis. GT-FAST provided a solution by enabling systematic, rapid, and automated sizing and synthesis of candidate F6 architectures. Its capabilities included:

- Mass, power, and cost budgets for each module and for the architecture as a whole
- Two-metric Pareto front for trade study capability
- Comparison against monolithic spacecraft (which has particular value for fractionated architectures)
- Creation of a Suite of Enumerated Architectures (SEA) and Suite of Enumerated Designs (SED)

Like EXAMINE, GT-FAST is Excel-based. Inputs to the tool are both discrete—i.e., which fractionated subsystems are in which module, and which modules are on which launch vehicle—and continuous—i.e., various specifications of the payload mass and power, orbital characteristics, etc (Lafleur and Saleh, 2009b), (Lafleur and Saleh, 2009a). Applied effectively, GT-FAST was a valuable tool for rapid assessment

during the concept phase. However, it is also noted that GT-FAST leaves out complexity analysis and does not model connections/interactions between either modules or components, and therefore the tool could still be expanded upon with new VDD methods.

Finally, Purdue University began building the Analytic Work Bench (AWB) with the development of several individual tools, such as for Robust Portfolio Optimization, Systems Operational Dependencies Analysis, Systems Developmental Dependencies Analysis, and Multi-Stakeholder Dynamic Optimization (Davendralingam et al., 2014). Separately, the Fractionated Satellite Design Space Exploration Toolbox (FS-DSET) began integrating flexibility and complexity models for evaluating fractionated spacecraft architectures (Neema et al., 2018). Collectively, these architecture modeling and analysis toolsuites provide the user diverse means of performing architecture-level tradespace exploration with VCA principles. However, limitations exist, such as how the tools were developed separately and how they lack a consistent taxonomy for space systems.

2.3.4 The Case for Purdue's Own Toolsuite

In 2019, Purdue University announced the Cislunar Initiative "aimed at accelerating the development of a cislunar region's economy" (Sequin, 2019). Considering that call to action, an integrated analysis environment would be extremely valuable for studies related to opportunities in cislunar space. Additionally, this environment would enable examination of the opportunities in OSAM concurrently with emerging space industry interest and initiatives. Drawing on the experience of various architecture building tools from industry, government, and academia, any new analysis environment would be most valuable if the tools are integrated for the user and are adapted specifically for space systems engineering. Finally, the environment should integrate the most recent VDD and tradespace exploration methods and leave easy means of including new, future methods.

CHAPTER 3. DEVELOPMENT OF MODELING ARCHITECTURES AND PARAMETERIZATION OF SPACECRAFT (MAPS)

After reviewing the state of system-of-system engineering, Value-Centric Analysis, and architecture tool suites in industry, government, and academia, a case was made for an advanced modeling and analysis environment at Purdue. This Chapter will review the purpose of a new environment and the methods employed. Section 3.1 outlines how the environment responds to identified needs. Section 3.2 explains the representation of space systems using network theory, which is central to the implementation of SoSE techniques. Section 3.3 explains the space systems modeling techniques. Section 3.4 explains the VCA analysis processes, which were selected following the literature review in Chapter Two. Section 3.5 explains how the environment is modifiable for future studies. Finally, Section 3.6 summarizes the environment's current capabilities and provides a full workflow diagram based on both the SoSE DAI process and the SEBoK decision management process.

3.1 Purpose

The Modeling Architectures and Parameterization of Spacecraft (MAPS) is a modeling and analysis environment developed to fill several needs. MAPS addresses the first research question:

> How can various existing tools for value analysis of space systems or SoS be integrated to improve decision support and to improve usability?

To answer this question, I hypothesized that by integrating five systems analysis tools—encompassing systems design, alternatives enumeration, flexibility, complexity, and life cycle cost—an environment could be created that enables improved evaluation

of space systems concepts. Thus, the MAPS environment was created, which is valuable because it can model emerging space systems concepts, it provides users access to a suite of SoSE and VCA tools, and it integrates the tools to improve decision support and increase usability across all studies.

First, a need was identified to enable holistic evaluation of space systems problems, especially understanding them as a system-of-systems problem (Guariniello et al., 2020). The establishment of the Purdue Cislunar Initiative in Fall 2019 reflected the need for analysis of missions to cislunar space (Sequin, 2019), and it was envisioned that MAPS would be utilized for such future cislunar opportunities. The possible problems, however, could range from Earth-orbit satellites to human exploration missions to unique space business ventures. For all space systems problems, MAPS was designed from the start with several key capabilities:

- Space Systems Design Tool, for creating mass and power budgets
- Launch vehicle analysis, with respect to the performance, volume, and cost of commercial launch vehicles
- Architecture Enumeration Tool, for enumerating architecture alternatives

In addition, for analysis of human missions, MAPS can model elements including crew, logistics, and Environmental Control and Life Support Systems (ECLSS) to build habitats. For problems encompassing a space SoS, MAPS can model stakeholders such as governments, customers, and companies and simulate interactions and economic models. Many human and robotic space systems problems can be investigated using these capabilities in MAPS.

Next, a need was identified to integrate tools for applying the best and most recent SoSE and VCA methods. A suite of tools was desired to respond to multiple scenarios and to trade a variety of value measures for decision support. To address this need, several value and cost tools that had previously been developed were selected for inclusion in MAPS, including:

- Flexibility Tool (Neema et al., 2018)
- Complexity Tool (Tamaskar et al., 2014)
- Additive Value Tool (Parnell, 2016)
- Life Cycle Cost Tool (Wertz et al., 2011)

Including these tools means that previous analysis can be extended, since they were helpful for certain types of analysis but were still disconnected tools with taxonomies that could change from one study to the next. Various improvements were made during integration of the tools into MAPS, including:

- adaptation for space systems,
- development and use of a consistent taxonomy,
- development of higher fidelity mass and power budgets,
- options for classes of space systems,
- more user-friendly software structure via object-oriented programming (OOP), and
- addition of new analysis scenarios.

In particular, it was chosen to standardize the use of OOP while developing MAPS in MATLAB. This improved the quality and modularity of the software tools. The ability to perform analysis in a single, integrated environment improves extensibility and repeatability for industry and government customers.

In summary, a call to action for evaluating space concepts using SoSE was presented in (Guariniello et al., 2020), and several specific needs were identified by reviewing architecture analysis tools in industry, government, and academia (in Section 2.3). MAPS fulfills these needs, and this environment will enable examination of emerging space industry opportunities and initiatives. Further, MAPS was designed to apply the decision support process, trading stakeholder values and identifying valuable design characteristics for further investigation and investment.

3.1.1 How does MAPS compare to existing methods?

It is important to note that although MAPS was created to fulfill several existing needs, it was not intended to change existing methods. That is, MAPS was created as a means of implementing the existing and up-to-date SoSE modeling and analysis methods in a single, integrated environment rather than in separately-developed tools. It facilitates existing space systems design practices but does not attempt to be a unique practice. This is reflected in the way that the workflow for MAPS follows the decision management process laid out by the SEBoK (from Section 2.2.2, Fig. 2.2). MAPS additionally uses traditional space systems design practices, such as those outlined in (Wertz et al., 2011). The value of MAPS over the many existing individual tools, therefore, is that MAPS *supports* current practice, makes it *easier* for users both to use the integrated tools and to integrate new tools, and provides a *more holistic* view of the decision support process.

3.2 Representation of Space Systems using Network Theory

Analysis in MAPS requires representation of space SoS using network theory. Various sources in literature have demonstrated the effectiveness of using a network representation when quantifying value, and the SoSE approach to modeling is closely coupled to this approach. Network theory, which is part of graph theory, involves the understanding of nodes and links in structural graphs. Structural graphs give meaning to the interactions between systems of an architecture. When two elements are linked in a structural graph, it means that one *interacts with* the other, and the link—an arrow icon in a visualized graph—is directional to represent some flow caused by that interaction. We can extend this concept to give mathematical meaning to value measures associated with space systems architectures. Since we know already which value measures are of interest to the stakeholders, the next step is to establish an ontology for representing space systems, and then structural graphs can be created.

A taxonomy with useful definitions was introduced in Chapter Two. Within the perspective of network theory representation, some of these definitions should be discussed further, since they have been developed specifically to reflect different levels of abstraction and system interrelations. Starting from a top-down perspective, an *architecture* encompasses all the elements of a mission. A space architecture will be defined by one or more free-flying *modules* or *assemblies* of modules. The modules can subsequently be defined by selecting required *subsystems*, including any payloads. Finally, the subsystems are designed by selecting the necessary *components*. This general flow is reflected in Fig. 3.1.

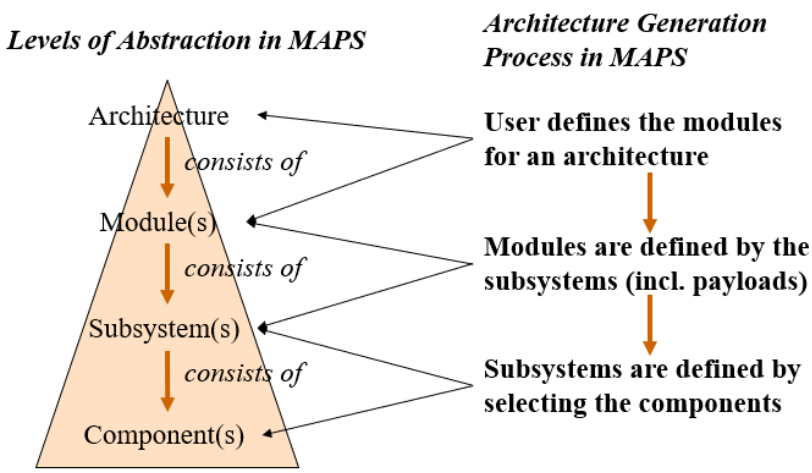


Figure 3.1: The MAPS process for defining an architecture, compared against the levels of hierarchy used in the MAPS taxonomy. In general, lower levels include a higher number of objects.

In MAPS, the process of selecting subsystems and components and creating structural graphs is performed by the Space Systems Design Tool. This tool is explained in more detail in a later section. However, it is important to understand how these different levels of abstraction impact the creation of a structural graph.

Components are at the lowest level of abstraction. A structural graph will be comprised of nodes, which are components, and links, which are the interconnections between components. Links are directional and represent the flow of electrical power, data, fluid, and/or thermal energy from one component to the other. In special cases,

links may also represent structural attachments; for instance, a motor is structurally linked to a solar panel. A notional structural graph for a spacecraft is shown in Fig. 3.2. A component-level graph is required input for evaluation of various value measures based in network theory.

Subsystems are defined as groups of components. Groups of nodes reflect system modularity. A common modularization scheme was decided for MAPS, rather than developing MAPS to be able to quantify or trade modularity. It is common practice in space systems engineering to allocate specific functions directly to a group of components and then to identify it as a "subsystem," e.g., propulsion or attitude control. Seven common subsystems are identified in MAPS through this functional allocation:

1. Thermal Management
2. Attitude Determination and Control Subsystem (ADCS)
3. Power Generation and Distribution
4. Propulsion
5. Communications
6. Avionics
7. Environmental Control and Life Support Subsystem (ECLSS)

This list closely matches the elements listed under Space Vehicle Bus in the Work Breakdown Structure (WBS) from NASA's Cost Estimating Handbook (NASA Executive Cost Analysis Steering Group and others, 2000), in the WBS from the the defense acquisition standard MIL-STD-881D (DoD, 2018), and in the WBS from *Space Mission Engineering: The New SMAD*, Ch. 11 (Wertz et al., 2011). These three sources differ from each other in only a few ways, so a couple changes were made for MAPS. For instance, the avionics subsystem is based on what is called "Flight Software" in all three sources and also combines the command and data handling components, which was separate in some of the sources and combined with the

communications subsystem in others. Also, "Tracking, Telemetry, and Command" is renamed simply the communications subsystem. Finally, although "Structures and Mechanisms" is considered a subsystem in NASA's Cost Estimating Handbook and MIL-STD-881D, structures is not considered a specific subsystem in MAPS, and any mechanisms may be included when relevant as a payload.

Payloads are additional subsystems that can be defined by the user. Payloads are generally considered separate elements in a WBS or cost estimate (NASA Executive Cost Analysis Steering Group and others, 2000), (DoD, 2018), (Wertz et al., 2011). They might be special versions of the common subsystems, e.g., a communications payload, or they might be unique science instruments. In MAPS, the structural graph for each payload should be provided by the user for appropriate fidelity in analysis. However, MAPS will generate a default graph for any payload if one was not provided. One default payload is shown on the example structural graph in Fig. 3.2.

Groups of nodes play a special role in graph theory, since groups effect the computation of certain metrics. A space subsystem fits the description as "an independent chunk that is highly coupled within, but only loosely coupled to the rest of the system" (Holtta-Otto and de Weck, 2007). This characteristic will impact the computation of coupling complexity, for example. The dashed boxes in Fig. 3.2 outline some examples of subsystems as groups of nodes on a structural graph.

A *module* is created by connecting one or more of these common subsystems. In general, the connections within a module would be established during manufacturing and assembly, and they are intended to be permanent. The links are the same as those already discussed—meaning one component is connected to another via a directional flow. Importantly, the links are still between components, although groups of components can be viewed from a higher level of abstraction as part of a subsystem. A user can therefore create both the component-level Fig. 3.2 and the subsystem-level Fig. 3.3a to represent the same module ("Module 1"); however, only the component-level graph can be used for computations.

In contrast, an *assembly* represents a temporary connection of modules—for example, docked supply vehicles at the International Space Station. Again, links within an assembly are still connections between specific components—for example, specific components of the NASA Docking System enable both structural attachment points and transfer of data, power, fluids, and astronauts between habitat modules. The subsystem-view and module-view graphs in Fig. 3.3 are roll-ups only for visualization purposes. These graphics help us understand the taxonomy used by MAPS and the design process.

MAPS was designed to track the interactions of the systems in a space architecture using this specific ontology. Several tools in the MAPS suite are based in the network representation. They are utilized to build and track each element, including creation of matrices used to represent structural graphs in the code.

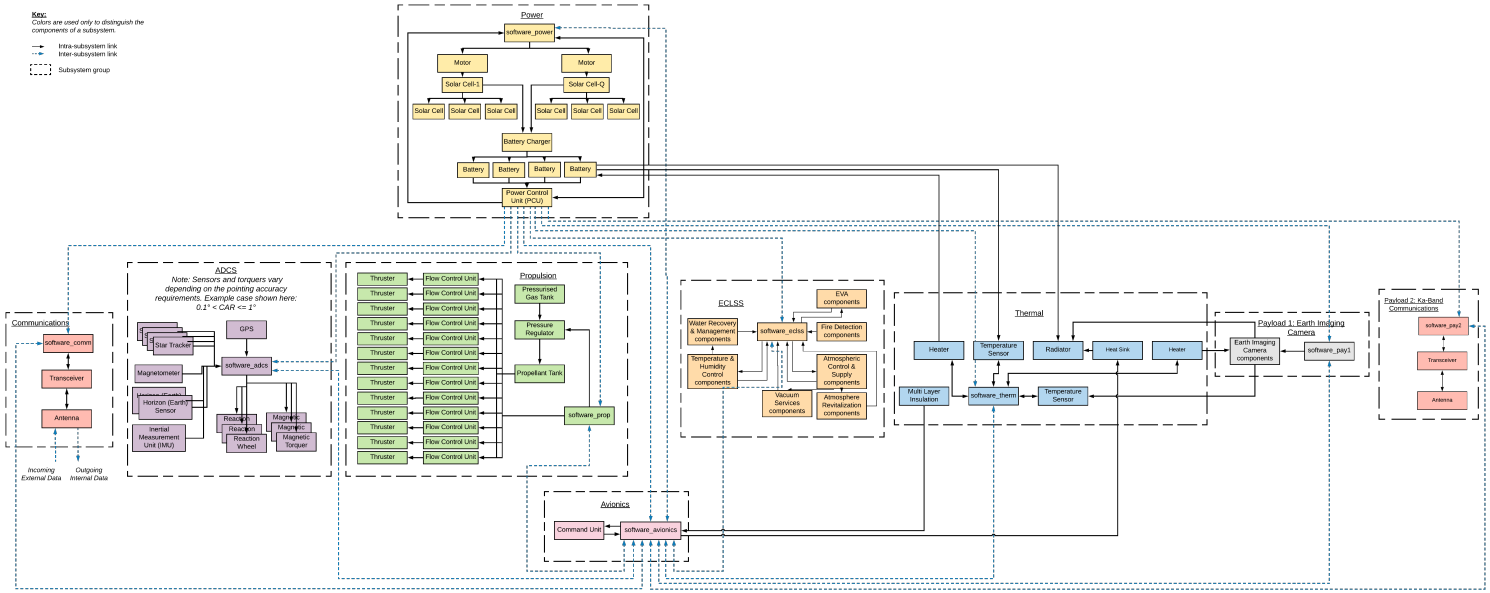
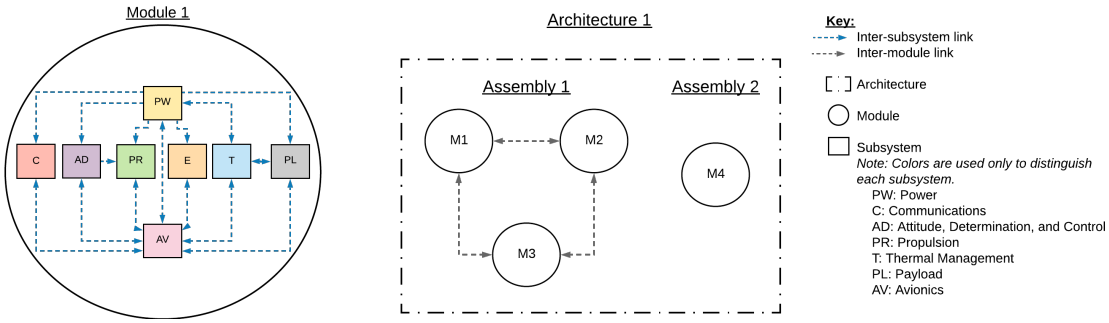


Figure 3.2: Notional component-level structural graph for a single spacecraft module.



(a) Notional subsystem-level graph for a module

(b) Notional module-level graph for an architecture, including two assemblies

Figure 3.3: Notional graphs at a higher level of abstraction, created for visualization only.

3.2.1 Representation in Matrices

Several matrix types are used in MAPS as an extension of the network theoretic representation of space systems. Matrices are useful because they are easily built, edited, and used in computations by the MATLAB coding environment in which MAPS was developed. The first and most important is the *adjacency matrix*, which sees widespread use in graph theory. This is a square nxn matrix that records the interconnections of n components. Since the adjacency matrix is built to directly represent the component-level structural graph, all the components are listed in the rows and columns, such as in the simple example in Fig. 3.4. In an adjacency matrix J_m for module m, each cell $j_{i,k}$ contains a "one" where the component of row i connects to the component column k. That is, the adjacency matrix in the the example can be read to say,"Component 2 connects to Component 1". If there is no linkage, the cell contains a "zero". Finally, the components are listed in a specific order of subsystems, the same as used for the module matrix.

Another potentially useful expression is the *module matrix*, which tracks the subsystems that exist in a given module. Previous authors have used various means of depicting module designs, such as small icons [Lafleur and Saleh] or a table with

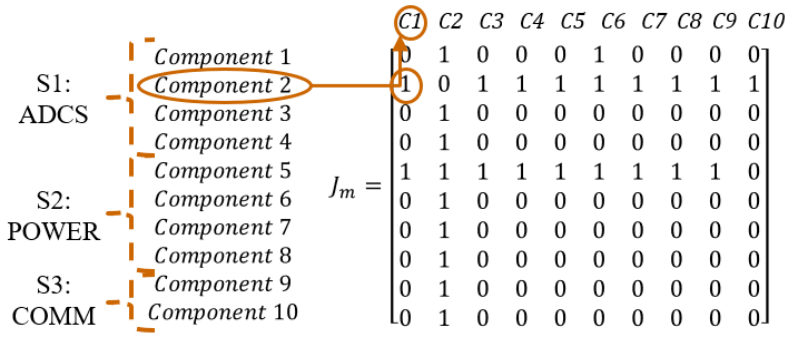


Figure 3.4: Notional adjacency matrix for a spacecraft module, including components associated with multiple subsystems.

ones/zeros representing the presence of subsystems [Neema]; however, a formalized matrix representation could be valuable. The module matrix is proposed here as a means of tracking a user's spacecraft design strategy.

The purpose of a module matrix in MAPS is shown when a user decides that an architecture should follow a certain design strategy. The strategy may be that all modules must have a specific set of subsystems (e.g., communications and power and propulsion), or alternatively, it may be that only a certain number of a given subsystem can exist (e.g., only one ECLSS is needed). There are two proposed variations of the module matrix. First, when each module is instantiated in the MAPS software, the module is assigned a single-column module matrix to track its own subsystems. Also, each architecture instantiation is assigned an extended module matrix which is a concatenation of all the columns created for its modules. By recording both variations, either can be used in MAPS analysis depending on which reduces computational time.

A module matrix is similar to an adjacency matrix, but it has seven rows for the seven common subsystems. These seven subsystems are listed in a default order—as mentioned, similar to the organization of an adjacency matrix. Optional payloads may add rows at the bottom; however, it is important to realize that the full list of user-defined payloads must be included. The full list is included so that the any given row index refers to the same payload across all modules. In the module matrix G_m

for module m, each cell $g_{n,m}$ contains a "one" if the subsystem of row n is present in the module. Additional columns in the extended module matrix represent separate modules m. An example module matrix is shown below in Fig. 3.5.

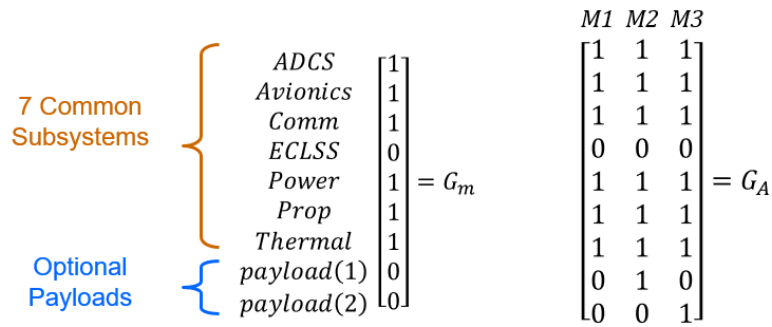


Figure 3.5: Notional module matrix for a spacecraft, including the single-module variation and the extended matrix for an architecture with multiple modules.

Finally, there may be architectures where it is important to track the movement of modules in multiple assemblies. This may be applicable for human space exploration missions or in on-orbit assembly operations, for example. The *assembly matrix* is proposed to serve in these cases and is created similar to the module matrix. The assembly matrix $H(t)$ has as many rows as there are modules in the architecture, and as many columns as there are assemblies. Each cell $h_{m,a}$ contains a"one" if the module of row m is present in the assembly of column a. An important difference here is that the assembly matrix can change over time and is therefore referred to as $H(t)$. This matrix representation should only be used if it can provide value for modeling and assessment of multiple changing assemblies. All three matrix options support the computation of value measures used in architecture assessment and trade-off studies.

3.2.2 Architecture Enumeration

Referring to the decision management process presented in the SEBoK (Fig. 2.2), "Generation of creative alternatives" is the second step, following "Develop objectives and measures" (Madachy et al., 2019). Additionally, Parnell reminds us that "the

generation of good alternatives is critical to identifying higher value" (Parnell, 2016). It is therefore essential that MAPS supports means for users to model various space mission concepts as well as enumerate architecture options for each mission concept.

Efforts to define different mission concepts is done first, whether within MAPS or externally using creativity/brainstorming methods (Parnell, 2016). One option for how this might be done for space systems concepts is through definition of one or more spacecraft design strategies. These can be described using module matrices and are input to the Space Systems Design Tool. In another option, the user may choose to define different sets of required/optional payloads or fractionatable elements before running the MAPS analysis tools. Any space systems concepts set up in MAPS can be passed on to the architecture enumeration tool and further to the architecture analysis of alternatives.

The representation of space systems using network theory directly enables architecture enumeration. If an architecture can be defined by the number of modules in it and the payloads composing each module, then all the architecture options can be enumerated by determining how many ways n distinct objects (payloads) can be placed in as many as n identical boxes (modules). In MAPS, the Architecture Enumeration Tool distributes the list of payloads and fractionatable elements into a range of modules. The result is a map of the module designs, each of which can be passed on to the Space Systems Design Tool to finish the definition of the architecture before trade-off analysis of the design alternatives.

3.3 Space Systems Design Tool

The Space Systems Design Tool in MAPS performs subsystem and component selection for individual modules, and it evaluates mass and power budgets based primarily on parametric relationships from historical space systems. However, the parametric relationships have also been supplemented with specific example data on commercially available hardware to improve the mass and power estimates. Finally,

it also builds the adjacency matrices critical to value analysis of the architecture. This process is guided by a collection of Subsystem and Component Selection and Connectivity Rules, which have been developed for MAPS similar to those employed in (Neema et al., 2018). These Rules guide the modeling process of a module's subsystems and components using OOP.

To begin, the Space Systems Design Tool needs to be passed information on the payloads on the given module and any desired design strategy from the user. Each payload must be defined by its dry and wet mass, power requirement, and pointing requirement. Additionally, a payload could have a unique adjacency matrix; otherwise, a simple, generic "Base Payload" adjacency matrix is built. Design strategies for a module are most easily passed into the tool via a module matrix, simply identifying which subsystems need to be included in that module. With these inputs, the tool takes over to select the subsystems and components that support the payloads and design strategy.

First, the Subsystem Selectivity Rules guide selection of subsystems to be included in the module, unless the user has input a design strategy which overwrites the defaults. For example, OSAM-centric or fractionated architectures may choose to exclude subsystems like communications or propulsion if the architecture is designed to have an assembly where other modules provide those functions. The Subsystem Selectivity Rules are:

- There are 7 Common Subsystems: ADCS, Avionics, Communications, ECLSS, Power, Propulsion, and Thermal
- Required subsystems in all modules: Power, Avionics, Communications
- Optional subsystems: ADCS, ECLSS, Propulsion, Thermal
- Additionally, payloads of any type may be added to a module or replace a default subsystem

Each selected subsystem is instantiated in using a class, which is a template built in MAPS for each subsystem—for example, "BasePropulsion". The specific subsystem instance is linked to the module instance, which is part of OOP practice. This allows for the specific designs of the subsystems to be tracked while also allowing many instances of "BasePropulsion" and other subsystem classes to be built.

Next, the tool steps through the Component Selectivity Rules, which have been developed for MAPS based on component libraries and configuration functions. In space systems, many subsystem design choices are interconnected with design choices in other subsystems, making component selection no simple task. For example, the design of the power generation components cannot be performed without knowledge of the total module's power requirements, but at the same time, the ADCS components cannot be designed without knowledge of the spacecraft total mass. Therefore, there is a linkage between the power required by the ADCS components and the mass of the power generation components. To address this complexity, the tool runs configuration functions to perform the component selections in a carefully defined process. An overview of the Component Selectivity Rules are provided in the Appendix in Table A.1.

During the components selection process, the subsystem instances are built up with additions to the dry mass, wet mass, and power requirement. These parameters are computed using a combination of historic parametric sizing guidelines from *Space Mission Engineering (SME)* (Wertz et al., 2011) and additional data from commercial hardware. *SME*—and the original published text, *Space Mission Analysis and Design (SMAD)*—has provided space systems engineers with sizing ratios and guidelines since 1999 (Larson and Wertz, 1992), and these preliminary values are appropriate for the architecture-level comparative assessments meant for the conceptual design phase. However, the author found it desirable to improve MAPS beyond the parametric sizing and to add as much fidelity as possible.

It should be acknowledged that *SME* guidelines, published in 2011, may not reflect the current space industry, a decade or more later. Since then, many in the industry

have commented on advancements to manufacturing processes, miniaturization, and the availability of commercial-off-the-shelf (COTS) spacecraft hardware. With this in mind, additional effort was made to find publicly available data on current state-of-the-art COTS hardware. Then, in various subsystems, specific components were detailed with mass and power requirements that match this industry data. For example, data has been added for lithium-ion batteries, CubeSat deployable solar arrays, monopropellant (hydrazine) thrusters, silicon-rubber flexible heaters, and more. This is an improvement over the *SME* parametric sizing guidelines, which are sometimes more general with respect to sizing of a whole subsystem and typically are based on mass percentages. These improvements are especially important for modeling small satellites, where components such as avionics boards or batteries can have significantly different sizes for CubeSats compared to traditional monolithic satellites. With the improvements, the MAPS Space Systems Design Tool is still only at about medium fidelity, appropriate for architecture-level assessment.

The final objective of the Space Systems Design Tool is to build adjacency matrices, which it does by applying the Component Connectivity Rules. These rules, presented in the Appendix in Table A.2, are split in two categories: intra-subsystem and inter-subsystem, as was done in (Neema et al., 2018). The improvements made for these rules in MAPS correspond with the improvements to the component libraries and configuration processes. The intra-subsystem connectivity rules describe how components within a system should be linked in the structural graph. The inter-subsystem connectivity rules describe any links from one subsystem to another, which is most often between the virtual "*software*" components—which are default additions to each subsystem (Neema et al., 2018)—and the power distribution components. An example component-level structural graph reflecting the many connections is provided in Fig. 3.2. Scripts developed for MAPS carefully build the appropriate adjacency matrices for each subsystem and then combined for the whole module, which completes the work of the Space Systems Design Tool.

3.4 Value-Centric Analysis in MAPS

To review, MAPS provides an environment in which a user can perform SoSE, can generate and assess design alternatives as steps in the decision management process, and can apply the VCA method to do the assessment. The theory behind VCA and the need for value measures during architecture analysis has been presented in Chapter Two. This section will describe the models used in MAPS for each architecture attribute, including selected value and performance measures, life cycle cost, and value functions.

3.4.1 Flexibility

"Flexibility" is used as a category for value measures, including adaptability, scalability, evolvability, and maintainability, as presented in Section 2.2.4. Table 3.1 defines these four value measures (adapted from (Brown and Eremenko, 2008)). Four scenarios (adapted from (Mathieu and Weigel, 2005)) are also provided in Table 3.1, which provide the context of each measure. Additionally, the definitions and analysis methods for adaptability and evolvability have been presented in previous work by the author (Grande et al., 2020), but the work will be expanded upon here with the addition of two new value measures within the flexibility category.

In MAPS, the adaptability scenario is analyzed using the Neema flexibility measure, which evaluates the possible architecture reconfiguration options that meet the new performance threshold (Neema et al., 2018). The scalability, maintainability, and evolvability scenarios are analyzed in a similar manner. For these three value measures, the Neema model is used to enumerate architecture options with the new or replacement subsystem/payload. Additionally, it is assumed in the adaptability scenarios that the architecture has not yet been launched, while it is intended in the scalability, maintainability, and evolvability scenarios that the architecture is in operation in orbit.

Table 3.1: Value measures in the flexibility category, with definitions (adapted from (Brown and Eremenko, 2008)) and scenarios for analysis (adapted from (Mathieu and Weigel, 2005)).

Value Measure	**Definition**	**Scenario**
Adaptability	the ability to reconfigure a system to meet new needs or circumstances	The architecture is required to meet a new performance/cost threshold.
Scalability	the ability to add components, capability, or performance to a system throughout its lifetime	The stakeholder wishes to add a new payload to add a capability to the architecture.
Maintainability	the ability to replace components that have failed or are near end of life	One subsystem/payload must be replaced.
Evolvability	the ability to replace and upgrade components due to technology obsolescence	The stakeholder wishes to replace a subsystem/payload to add capability to the architecture.

The measure of flexibility from (Neema et al., 2018) builds off the network representation of space systems and therefore requires an adjacency matrix for each module. As explained in Section 3.2, this matrix must be at the component level, where components are represented as nodes and their interconnections are represented as directed and weighted links. It can then be said that a system is "flexible" if new nodes can easily be added and connected to the other nodes while still maintaining acceptable performance (or value). This model was originally developed for fractionated spacecraft architectures, wherein spacecraft functions are distributed between some number of free-flying modules that collectively work together to achieve mission objectives (Neema et al., 2018). However, the process can also be applied to any architecture, as will be demonstrated in this thesis.

In the first step of the Neema flexibility measure, the user-defined list of fractionated subsystems (including payloads) is passed into the MAPS Architecture Enumeration Tool, which defines a finite trade space of physical architecture design alternatives. Architecture designs are characterized by the number of modules and the distribution of fractionated subsystems in each module. Although some users may have already defined the desired physical architecture and may only want to evaluate flexibility for the baseline, architecture enumeration is still required. The variable TD equals the number of total number of design alternatives and will be used later. Next, the MAPS Space Systems Design Tool is employed to build the architecture around the fractionated subsystems of each module and to define the adjacency matrices.

The next step is to count how many design changes, DC_{r-t}, are required to move from the reference architecture r to each other alternative t on the list. Design changes are defined by the addition or removal of either nodes or links, as in Eq. 3.1. To find DC_{r-t}, the flexibility tool was carefully designed to compare the architectures' adjacency matrices. "Acceptable" designs are then defined as those that have less than a user-defined maximum number of design changes. When considering that given reference, the number of design changes is recorded for each alternative, and the alternatives that are not "acceptable" can be removed from the list going forward.

$$\begin{aligned} DC_{r-t} = \text{(number of node additions)} \ + \ \text{(number of node removals)} \ + \\ \text{(number of link additions)} \ + \ \text{(number of link removals)} \end{aligned} \tag{3.1}$$

Next, the definition for "feasible" designs is based on the normalized change in performance or value. A common performance metric might be initial mass in LEO (IMLEO), for example. The total architecture mass can easily be referenced in MAPS from the output of the Space Systems Design Tool. Though Neema's model described performance change, where performance was total mass (Neema et al., 2018), it would be equally valid to substitute any other value measure. After finding the performance of each alternative, the performance change ΔP_t is recorded. There are two options:

(1) performance change from moving from the reference architecture to the given alternative architecture, or (2) the difference between a user-defined performance threshold and the alternative's performance, as shown in Eq. 3.2.

$$\Delta P_t = P_t - P_{ref} \tag{3.2}$$

A user may choose to define a performance threshold if, for example, the total architecture mass must be below the maximum payload capacity of a certain launch vehicle. It is important here to acknowledge a nuance when using mass as the performance measure. For space systems, *less* mass would be considered *higher* performing, which can be confusing when discussing performance. Two options to manipulate the performance measure include multiplying the change in performance by -1 to switch the sign, or subtracting the mass from a constant, as shown in Eq. 3.3. By manipulating performance in this way, an architecture mass *lower* than the reference will have a *positive* performance change.

$$P_t = Constant - Mass \tag{3.3}$$

Next, the performance change is normalized with respect to the maximum performance change present amongst the alternatives, as shown in Eq. 3.4:

$$NP_i = 180\Delta P_i/(max|\Delta P|_{t=1}^{TD} + 1) \tag{3.4}$$

where $NP_i \in [-180°, 180°]$. "Feasible" designs have a normalized performance between 0° and 180°. The list of design alternatives is further reduced by removing any alternatives outside the feasible range for normalized performance. Finally, the flexibility of the reference architecture is defined as the ratio of the number feasible and acceptable design alternatives (FD) to the total design alternatives (TD):

$$F = FD/TD \tag{3.5}$$

As already mentioned, the steps toward evaluating flexibility should be repeated for each design alternative, where each is considered as the "reference" in turn. Additionally, an improvement has been made to the Neema model for flexibility where it is only used in specific ways based on the operational context for a specific flexibility-type measure. For this improvement, the model has been adapted based on the scenarios defined in Table 3.1. For example, a new performance threshold is defined in the adaptability scenario, or in the scalability scenario, a new payload is added prior to architecture enumeration. The flexibility measure presented by (Neema et al., 2018) can demonstrate the scale of design reconfiguration and the impact to architecture-level value, but with the added context, the measure is now more effective at comparing architecture alternatives and provides insight for decision making.

One concern with the Neema model is that it does not reflect what design changes mean from a cost or complexity perspective (Grande et al., 2020). To address this concern, the model can be further adapted to substitute the change in life cycle cost in place of change in performance. Alternatively, the model can be adapted to substitute the normalized change in complexity. Future work may use the MAPS Flexibility Tool to explore these changes.

3.4.2 Complexity

The complexity model in MAPS was founded on the work of (Tamaskar, 2014), as presented in Section 2.2.5. Specifically, this model evaluates the design and development complexity experienced by the engineering design team. With this definition of complexity, Tamaskar used network theory to evaluate the measure and proposed that the most important aspects of complexity include coupling, directionality, and feedback loops. The component-level adjacency matrices created by the MAPS Space Systems Design Tool can therefore be employed for evaluation of architecture complexity. The following steps are adapted from (Tamaskar, 2014) and have previously been presented in (Grande et al., 2020).

Once the architecture is represented in a collection of directed, weighted structural graphs or adjacency matrices, each module is evaluated separately. The first step for the complexity measure is to redefine the directed links to account for indirect coupling. Initially, all weights are set equal to one, with the reasoning that there is not enough information for space systems to otherwise define link weights (Tamaskar, 2014), (Neema et al., 2018). Then, a list of all possible paths in the graph is generated, and the frequency that each link is used within the graph is calculated. Each link weight is redefined by multiplying the weight by its frequency.

In the second step, the coupling complexity of a module is calculated using Eq. 3.6:

$$C_{CC} = \sum_{f=1}^{F}(l_f \sum_{i=1}^{l_f} W_{i-f}) + \sum_{k=1}^{m} W_k \tag{3.6}$$

where F denotes the number of feedback loops existing in the graph, l_f denotes the number of links in loop f, W_{i-f} denotes the weight of link i in loop f, m denotes the number of links which are not part of a loop, and W_k denotes the weight of link k that is not part of a loop.

Next, the method next considers modularity and integration complexity. Each module is composed of N integrated subsystems, and the coupling complexity of each subsystem n is calculated C_{CC_n}. The same method for Eq. 3.6 is repeated, except now only a subsystem's isolated adjacency matrix is used. The module's integration complexity C_I is defined:

$$C_I = C_{CC} - \sum_{n=1}^{N} C_{CC_n} \tag{3.7}$$

(Tamaskar, 2014) proposed that design and development complexity for a module is actually *reduced* through the hierarchical organization of the subsystems, so he proposed a coefficient of integration α_I to reduce the module's complexity measure. Through his definition, $\alpha_I < 1$ is always true. The modifier coefficient of integration and the modified integration complexity are defined:

$$\alpha_I = C_I / \sum_{n=1}^{N} C_{CC_n} \tag{3.8}$$

$$C_{Im} = \alpha_I C_I \tag{3.9}$$

The final module complexity for a module m decomposed into N subsystems is therefore:

$$C_{MC_m} = \sum_{n=1}^{N} C_{CC_n} + C_{Im} \tag{3.10}$$

MAPS will record the complexity of each module in the architecture. However, the full architecture is integrated from M modules. These modules may be physically linked (in assemblies) or have only communications links with the ground control center(s) or other modules. Following the same approach as the complexity model for a decomposed module, the architecture complexity can be defined:

$$C_A = \sum_{m=1}^{M} C_{MC_m} \tag{3.11}$$

In future work, architecture-level integration complexity could be defined to modify this sum, similar to module integration complexity. Further research would also be required to determine if the design and developmental complexity needs a modifier. Any modifier would need to be backed by current practice for space systems design and operations. Alternatively, there is the option to include assembly-level integration complexity only if the modules are linked in an assembly, or alternatively to always include an architecture-level integration complexity factor when an architecture has more than one module. These options should be studied for future work.

3.4.3 Other Value Measures

Various measures other than flexibility and complexity could also be used to describe the value of an architecture. It is likely that a user performing analysis of alternatives in MAPS will have a list of value measures to evaluate. MAPS was therefore designed to be a modular software environment that a user can adapt to suit a variety of projects by integrating new tools.

New tools from SoSE and VCA might be integrated in the future for evaluating additional measures. For example, tools for system operational dependencies (Guariniello and DeLaurentis, 2017), system developmental dependencies (Guariniello and DeLaurentis, 2013), and resilience (Maghareh et al., 2019) could be added to MAPS. On the other hand, a generic Performance Tool was developed so that an analyst can add simple scripts that measure performance based on physical system attributes. These measures might include, for example, total mass, communications bandwidth, communications coverage, or number of customers served. A couple mass-based measures, as presented in Section 2.2.3, are already included in the Performance Tool. The number of value measures that can be recorded for a given architecture is unlimited, so there is a continuous path forward for MAPS improvement.

3.4.4 Life Cycle Cost

Cost may be a major focal point for most stakeholders; however, it must be treated separately from the value measures during architecture analysis of alternatives, (Parnell, 2016), (Brown and Eremenko, 2008). Cost is not assimilated into the additive value function, but analysis results should display graphs of value vs. cost. The Life Cycle Cost model in MAPS includes DDT&E, production, launch, and operations costs, as presented in Section 2.2.6. To estimate the cost of DDT&E and production (D&P), three models are available in MAPS: AMCM, USCM, or NICM.

The Advanced Mission Cost Model can be chosen to estimate D&P cost C_{DP} of space systems for human missions. It uses a single, multi-parameter function:

$$C_{DP} = (5.65 * 10^{-4})Q^{0.59}M^{0.66}80.6^{S}(3.81 * 10^{-55})^{1/IOC-1900}B^{-0.36}1.57^{D} \tag{3.12}$$

where C_{DP} denotes total development and production cost (in FY1999 millions of dollars), Q denotes quantity for production, M denotes system mass (in pounds), S denotes the specification number (specific to type of system, e.g. human habitat or planetary lander), IOC denotes expected operability year, B denotes the design generation, and D denotes the difficulty rating (Larson and Wertz, 1992), (Jones, 2015). The result is converted automatically into FY2020 dollars by MAPS. A cost matrix of these parameters has been created and is stored in the Space Systems Design Tool for each subsystem using best-guess estimates informed by literature and industry knowledge. If a user selects to use the AMCM, the architecture development and production cost is evaluated by sequentially evaluating the costs of the subsystems contained in each module of the architecture.

The Unmanned Space Vehicle Cost Model uses a set of CERs for non-recurring and recurring costs depending on the subsystem type. For example, the non-recurring cost of the Attitude Determination and Control Subsystem (ADCS), which includes development and one qualification unit, can be estimated as:

$$C_{NR} = 324M \tag{3.13}$$

where C_{NR} denotes the non-recurring cost (in FY2010 thousands of dollars) and M denotes the mass of the ADCS (in kilograms). The USCM defines recurring costs that include the manufacturing of the first flight unit, for example for the ADCS:

$$C_R = 795M^{0.593} \tag{3.14}$$

where C_R denotes the recurring cost (in FY2010 thousands of dollars). So if there will be one ADCS unit, then the D&P cost for the ADCS includes the sum of Eqs.

3.13 and 3.14. The total D&P cost of the space system would be equal to the sum of its subsystem costs as well as certain additional categories defined by the USCM, including Spacecraft Integration, Assembly, and Test plus program-level costs (Wertz et al., 2011). In MAPS, if a user selects to use the USCM, the architecture is evaluated by sequentially evaluating the costs of the subsystems contained in each module of the architecture and then adding these additional categories. The result is converted automatically into FY2020 dollars. A full set of the USCM's CERs can be found in the *Space Mission Engineering* text (Wertz et al., 2011) or on USCM Online (US Air Force, Space and Missile Systems Center, nd).

To find the cost of the payloads, the NICM can be used. The NICM uses a set of CERs depending on the instrument category to estimate the non-recurring cost of development plus one flight unit. For example, for an Optical Earth-Orbiting Payload:

$$C_{NR} = 1163 M^{0.328} P^{0.357} DR^{0.092} \tag{3.15}$$

where C_{NR} denotes the non-recurring cost (in FY2010 thousands of dollars), M denotes the mass of the instrument (in kilograms), P denotes the maximum instrument payload (in Watts), and DR denotes the total data rate (in kilobits per second) (Wertz et al., 2011). However, the D&P cost for the payload to be included on the full system must also include other categories, similar to the USCM, such as Integration and Test and management. CERs for these extra categories are provided in (Wertz et al., 2011). In MAPS, the instrument category for each payload should be input by the user along with the other payload properties, and then the NICM is automatically used in conjunction with the USCM. The full set of the NICM's CERs can be found in the *Space Mission Engineering* text (Wertz et al., 2011).

The other two pieces of the Life Cycle Cost model are operations and launch costs. The operations cost of the architecture is evaluated using the C_{DP} results and the user-defined mission duration as inputs. As explained in Section 2.2.6, the annual operations cost when using the AMCM can be estimated simply as 10% of the D&P

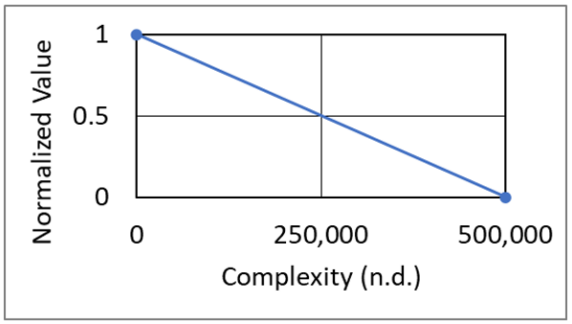
cost from the AMCM (Jones, 2018). The results provide a comparable, albeit low-fidelity, estimate for the architectures. When using the USCM, specific CERs are used to estimate use of aerospace ground equipment, launch operations, and orbital support. The total operations cost, included in the life cycle cost, is simply the annual costs multiplied by the expected number of years of operation.

Finally, the launch cost is added. Publicly available price and performance data for the SpaceX Falcon 9 and Falcon Heavy have been input to MAPS (SpaceX, 2020). An evaluation of the number of launch vehicles that are required is based on the vehicles' payload-to-orbit capacity. Additional launches required by simulated operational scenarios, such as the scenarios for value measures in the flexibility category, may also be included. The sum of the launch vehicle prices, total operations cost, and total DDT&E and manufacturing cost forms the life cycle cost for the architecture that can be used in the analysis of alternatives.

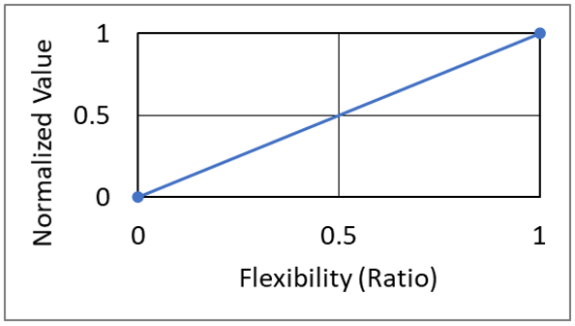
3.4.5 The Additive Value Model

Section 2.2.2, Eqs. 2.1-2.2, presented the final step of synthesizing results, in which value measures should be combined in the additive value model. In summary, the value measures data for a given architecture are normalized (on a scale from 0-1) using a set of value functions and are multiplied by swing weights. The normalized, weighted sum represents the single composite result for value.

The MAPS user must formulate value functions for each value measure by inputting the minimum acceptable value and the ideal value, which are the two defining points on the value function (Parnell, 2016). Additionally, an understanding of each value measure is required to decide the direction and shape of the curve. Unless otherwise specified, MAPS defaults to a linear value function. Examples are provided for the complexity and flexibility category measures in Fig. 3.6. Value functions are created for the case study in Section 4.2.3 (Tables 4.4-4.5). The value functions can be defined in an Excel spreadsheet which is input to MAPS.



(a) Normalized value vs. complexity



(b) Normalized value vs. flexibility

Figure 3.6: Example linear value functions to normalize selected value measures on a scale from 0-1.

The MAPS user must also input swing weights for each value measure. The development of swing weights is described in literature, for example, through creation of a swing weights matrix (Johnson et al., 2013), (Parnell, 2016). A swing weights matrix is created for the case study in Section 4.2.3 (Tables 4.6-4.7). The swing weights can be defined in an Excel spreadsheet which is input to MAPS.

After the user has defined these inputs, the Additive Value Tool can be called and results recorded for a given architecture. Finally, the results can be visualized in tables, graphs, and Pareto frontiers. The MAPS user is then capable of communicating trade-offs and supporting decision-making.

3.5 Enabling Extensibility and Repeatability for Future Studies

Central to the purpose of developing MAPS was to enable extensibility and repeatability for many years of future studies. The emphasis on extensibility means that future developers will be fully able to extend the existing capabilities, adding new tools and classes. The key to extensibility was the standardization of OOP and an organized, hierarchical structure to the many software files. Additionally, emphasis was placed during development on developing a common taxonomy, and then modifying the value models to be applicable to the widest range of space systems problems.

These solutions combined increase the range of problems that are within MAPS' scope and increase a user's ability to customize MAPS to explore the problems.

The emphasis on repeatability was important because the MAPS developers wanted future users to be able to easily navigate the tools and feel confident in the quality of the analysis. Having all the tools integrated in a single environment where user inputs are passed in for architecture analysis is only the first step in achieving repeatability. The most important part was actually developing a MAPS User Guide. This User Guide includes explanations of the taxonomy, how the tools work, and how to navigate the environment. Guidelines for creating "runner" scripts, which format user inputs and call the tools, were especially important to include. These guidelines explain when certain steps need to be done in a certain order, when certain inputs are required, and how various variables are formatted in MAPS. Example pages of the Guide are displayed in Fig. 3.7. The Guide is included in the Git repository for any user to access when they download the MAPS Git repository with all the code, and it is a critical supplement to the many code comments and example runner scripts. The Guide itself will continue to grow over time with the expanding application of MAPS.

3.6 MAPS Capabilities Summary

This chapter has presented an overview of the purpose and current capabilities of MAPS. A summary is provided below. A full workflow diagram of the steps to use MAPS for the decision management process is presented in Fig. 3.8. This diagram was carefully created based on the references for the SoSE DAI process (DeLaurentis, 2005) and the decision management process (Madachy et al., 2019), (Parnell, 2016).

- MAPS provides an environment in which a user can perform SoSE, can generate and assess design alternatives as steps in the decision management process, and can apply the VCA method to do the assessment.

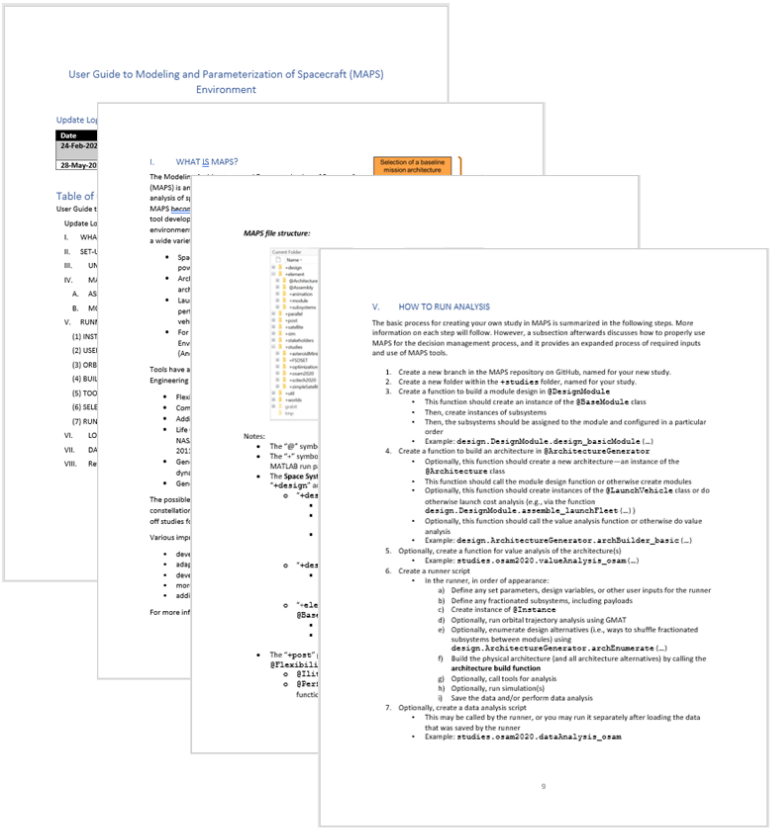


Figure 3.7: Sample pages from the MAPS User Guide, depicting an introduction to MAPS, the file structure, and how to run analysis.

- The MAPS environment addresses the key needs to evaluate emerging space systems concepts, to use the newest systems analysis tools, and to perform analysis in a single, integrated environment.
- The problem scope of MAPS is wide, including satellites, human exploration elements, and space systems-of-systems (SoS).

- Space systems are represented using concepts from network theory; key to this representation is the creation of structural graphs and matrices that can be used for later computations.

- A user has access to a suite of tools to perform non-traditional analysis of space systems architectures and to perform trade-off analysis in support of decision making. In particular, architectures are evaluated for value and cost, and results can be graphed on Pareto fronts.

- The Space Systems Design Tool models the subsystems and components of space systems by combining parametric sizing and commercial product information. This tool outputs mass and power budgets and adjacency matrices.

- Several tools were integrated in MAPS to evaluate value measures. The tools for flexibility and complexity were based on previous research but have been modified and improved for the modular MAPS environment, to match a consistent space systems ontology, and to apply to diverse space systems problems.

- The Life Cycle Cost tool evaluates the DDT&E and manufacturing cost using the Advanced Mission Cost model and then evaluates operations and launch costs at an acceptable level of fidelity for architecture analysis of alternatives.

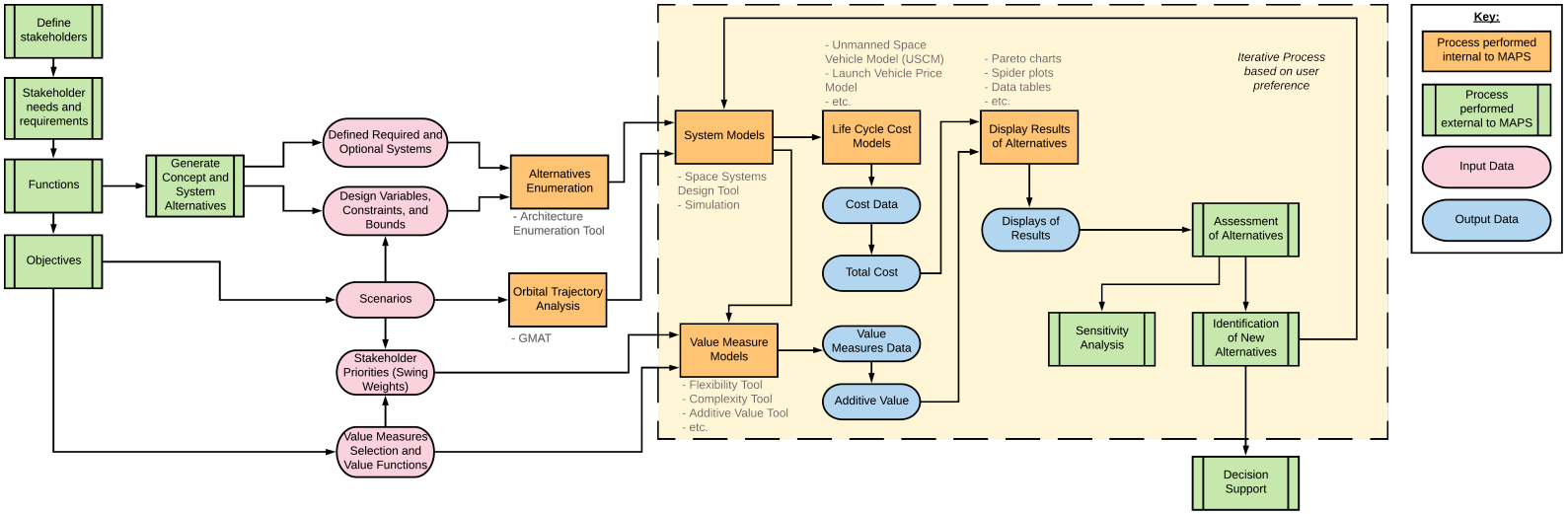


Figure 3.8: Workflow diagram of the MAPS environment for use in the decision management process.

CHAPTER 4. CASE STUDY: PERSISTENT PLATFORM WITH OSAM

A case study has been developed that will demonstrate the capabilities of the MAPS modeling and analysis environment and to respond to future opportunities in On-Orbit Servicing, Assembly, and Manufacturing. Therefore, the case study is designed to answer the second research question:

> What design features for a persistent OSAM platform, capable of hosting customer payloads can be varied to best balance value (e.g., flexibility, complexity) and life cycle cost?

Chapter Four will demonstrate an application of the previously described DAI process for SoSE while implementing the chosen method of VCA for analysis. Section 4.1, on Definition, includes definition of the decision opportunity, decision scope, and operational scenarios for the case study. Section 4.2, on Abstraction, includes identification of the main actors, networks of interactions, and the modeling approach for the case study. Section 4.3, on Implementation, describes how the model of the actors and network have been instantiated in MAPS as well as the study results. Finally, a brief sensitivity analysis on the parameters of the model will be presented in Section 4.4.

4.1 Problem Definition

During the Definition Phase described by (DeLaurentis, 2005), the SoS operational context, status quo, barriers, and taxonomy should be described (see Fig. 2.1). These initial products were presented in Chapter One for opportunities in a future OSAM market. With this research completed, a table can be created which outlines system categories, including Resources, Operations, Policies, and Economics (ROPE), and

their levels of organization (DeLaurentis, 2005). The ROPE table in Table 4.1 was created for an envisioned OSAM market. The table helps us understand the categories of systems involved in the SoS of interest and also begin to consider the scope of a potential decision opportunity.

4.1.1 Defining the Decision Opportunity

The decision opportunity is defined by the research question:

> What design features for a persistent OSAM platform, capable of hosting customer payloads can be varied to best balance value (e.g., flexibility, complexity) and life cycle cost?

However, this is only a loose definition of the opportunity so far. The concept must be defined in more detail before the decision opportunity can be defined in more detail. Early identification of the concept, functions, and operations are among the first steps in the MAPS workflow, as shown in Fig. 4.1(a). The workflow shows that these external products should be created before analysis in MAPS begins.

It was decided that the case study would focus on a persistent platform concept with OSAM capabilities that would host customer payloads in LEO. More detail on the concept of operations was laid out in a functional architecture, as shown in Fig. 4.2. Because this case study is centered on the idea of an OSAM market, it was also considered interesting to include certain repair operations:

- The power subsystem must be replaced on the platform
- Either a replacement power subsystem, a replacement module, or a full replacement platform is delivered—depending on the platform's servicing or assembly capabilities

Additionally, scalability operations were included:

- Stakeholder defines an increase to the total payload mass and total power to be supported by the platform

- Additional support subsystems are delivered to or manufactured on the platform, then installed on the platform
- Additional payloads are delivered to and installed on the platform

These operations form a scenario to explore the impact of OSAM capabilities. Operational scenarios will be discussed in the following subsection.

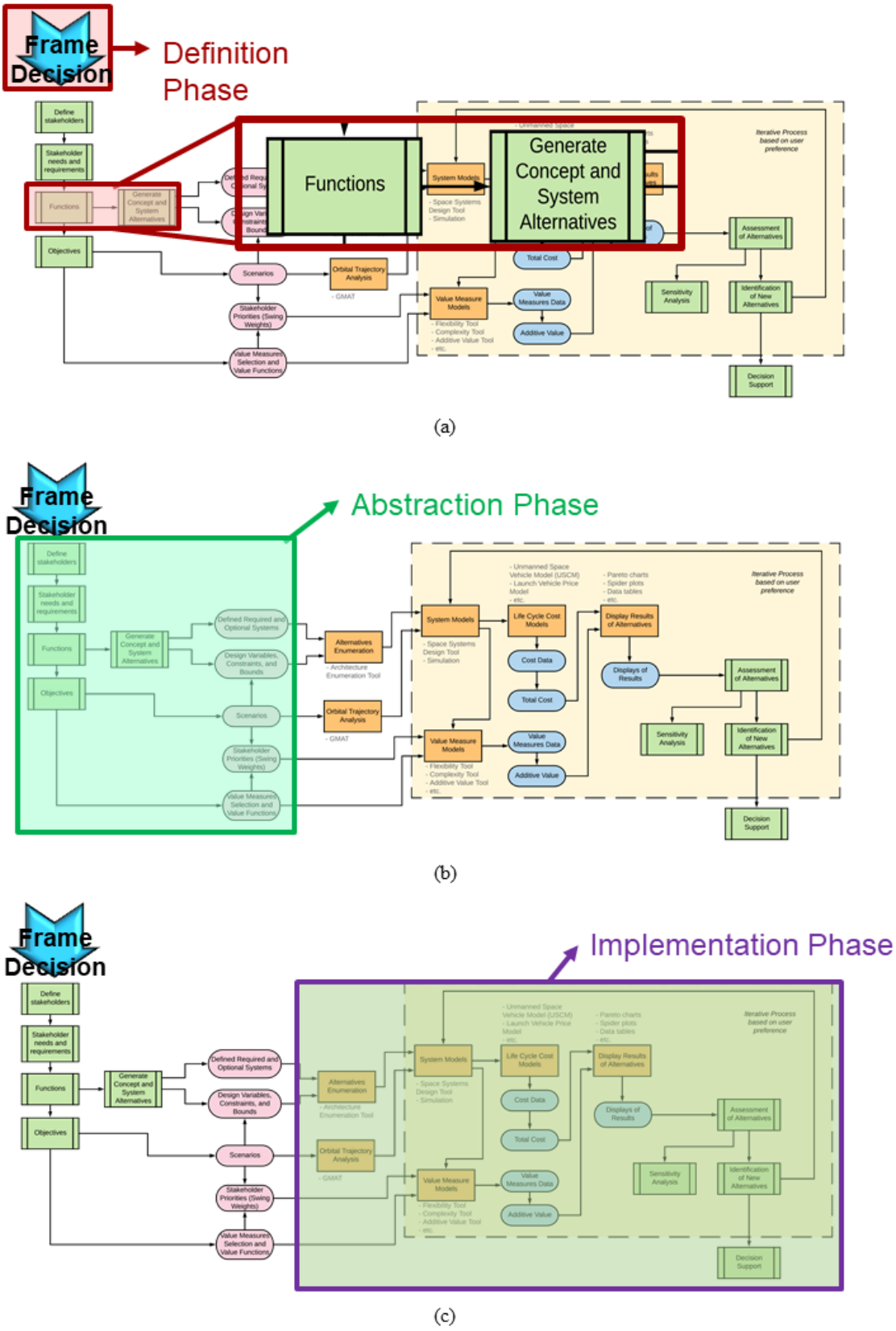


Figure 4.1: Highlighted sections of the MAPS environment workflow diagram during (a) Definition phase, (b) Abstraction phase, and (c) Implementation Phase.

Table 4.1: ROPE Table for a persistent platform in a space system-of-systems.

Level	Resources	Operations	Policies	Economics
α	• Scientific payload • Robotic arm • Power subsystem • Propulsion subsystem • Attitude control subsystem • Launch vehicle • Ground station employee • Spacecraft/robotics manufacturing employee	• Command, control, and data handling for a science payload • Command and control of a robotic arm • Launch operations and maneuvering of a launch vehicle • Manufacturing of a spacecraft/robot	• Manufacturing standards for a spacecraft • Manufacturing standards for a launch vehicle • Communications regulations in LEO	• Cost to manufacture one payload • Cost to manufacture one launch vehicle • Cost to launch one payload • Price of a communications license
β	• Collection of science payloads • Fleet of launch vehicles • Teams of ground station employees	• Command, control, and data handling for a collection of science payloads • Command, control, and data handling for a resource management and manufacturing system in orbit • Managing a team of ground station staff	• International agreements concerning interference with space assets of other actors • Interface standards between payload and launch vehicle • Policies concerning selling services or products in orbit	• Cost to operate and manage a ground station • Cost to manufacture a system in orbit
γ	• Network of spacecraft in Earth orbit • Persistent platform company • Launch vehicle company • Federal Communications Commission (FCC)	• Operating a long-duration platform in LEO • Managing a persistent platform company and payload bookings • Managing a launch vehicle company • Managing communications licenses for satellites in Earth orbit	• Policies concerning company liability for space assets • Orbital debris regulations	• Ground operations cost to manage orbital network • Customer price to host payloads on a persistent platform
δ	• Global governing body for in-space operations and standards (e.g. UN COPUOS) • Solar system-wide network of spacecraft and service vehicles	• Operating a solar system-wide network of spacecraft	• Policies concerning selling services or products in the solar system	• Economics of global space industry

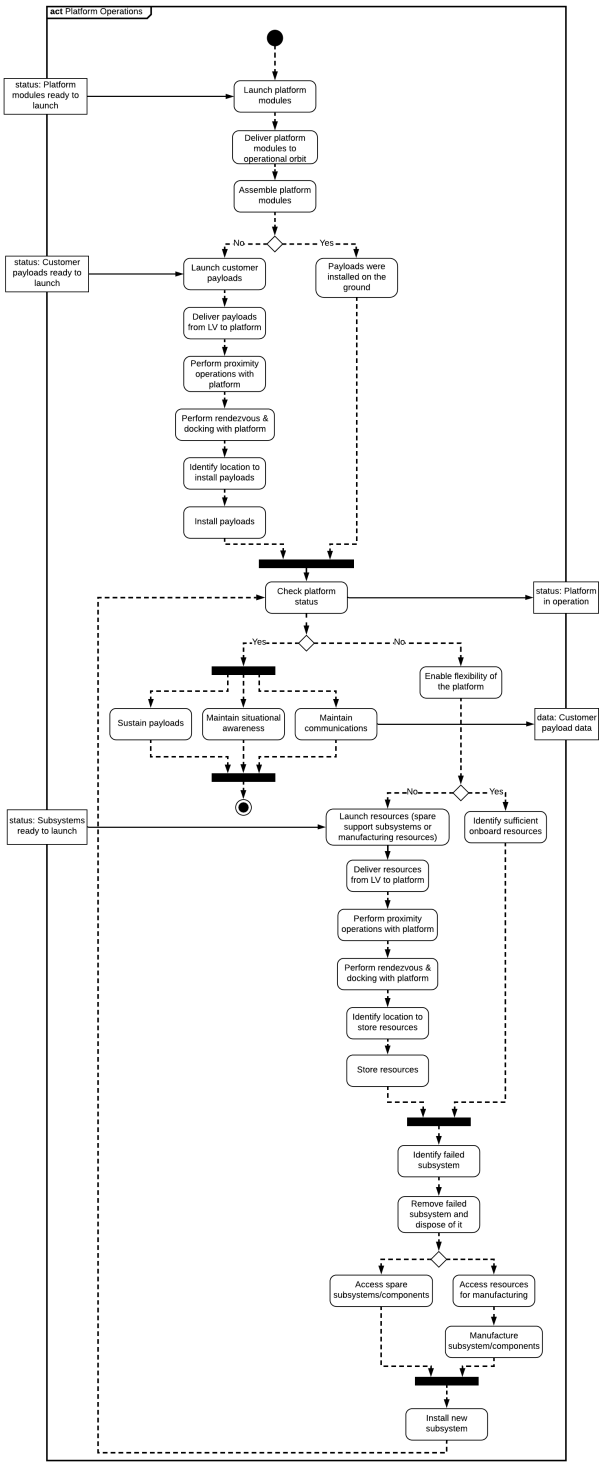


Figure 4.2: Functional architecture for the persistent platform concept.

Next, a morphological matrix was created to respond to the functional architecture. A number of creative methods can be used to generate designs (Cross and Roy, 2000), (Ullman, 2003). A morphological matrix was the method chosen for this case study, and it is used to generate sets of design alternatives specifically for each function (Cross and Roy, 2000). Tables 4.2 and 4.3 were created for the concept of operations and for the scalability scenario operations, respectively. The matrices show that there are many options for design decisions that can be within and out of scope.

The decision hierarchy (Fig. 4.3) is used to constrain the scope of the decision opportunity (Parnell, 2016). All of the questions or statements in the decision hierarchy are encompassed by the research question but are more specific. At the top are high-level decisions that are taken as a given and are therefore out of scope:

- The mission is to sustain customer payloads in LEO (550 km altitude).
- Payloads will be launched to LEO on a single vehicle option (SpaceX Falcon 9).
- The company's country of origin has a supportive regulatory environment.
- The company will develop subsystems to sustain the customer payloads.

In the middle of the decision hierarchy are decisions that are part of the decision opportunity:

- How much mass of hosted customer payloads should be supported on the platform?
- What OSAM capabilities should be installed?

Finally, at the bottom are decisions that can be deferred until later and are therefore out of scope:

- Specific decisions for component selections—e.g., communications frequency or propellant type

- How will data be controlled, and who will own/operate ground stations
- Customer price to host payloads and profit model

As a result of the decision hierarchy, the decision space was defined by a three variables: total supported mass of hosted payloads and presence of servicing and manufacturing capabilities. It was then clear that architecture alternatives should be generated in MAPS that combine the design variables.

Figure 4.3: Decision hierarchy, outlining the questions that are within and outside of the decision scope for the case study.

It was assumed that future design teams could trade the other higher-level or lower-level decisions from the hierarchy. Since those decisions were out of scope for this study, they defined the fixed parameters for all architectures, as in those parameters that are fixed and unchanging. Returning to the morphological chart, the full list of fixed parameters was compiled, for example:

- The platform would be designed with traditional spacecraft subsystems to support the customer payloads, e.g., solar power generation and Ka-band antennas for communications.
- All modules would be launched at the same price on a per-kilogram basis.
- A Service Module will be designed to loiter in LEO, to deliver all modules from the altitude achieved by the launch vehicle to the platform location, and to perform rendezvous and docking with the platform.

The full list of fixed parameters can be found in Appendix B. Returning also to the ROPE table, it was observed that the modeling scope was constrained to the α and β level for R,O,P, and E.

4.1.2 Operational Scenarios

Finally, the last product of the Definition Phase was operational scenarios. (Parnell, 2016) explains that the purpose of operational scenarios is "to understand how a system will operate in different environmental conditions." In addition, different objectives and/or priorities can be introduced in scenarios. In some studies, scenarios have been used to provide additional context for certain value measures, such as for flexibility category value measures (Mathieu and Weigel, 2005), as introduced in Section 3.4.1. For example, a Scalability Scenario might prioritize value measures such as "ability to upgrade system performance," "cost to upgrade," and "scalability," and it might provide context for the "scalability" value measure. This was the justification for introducing the Scalability Scenario into the case study.

This study had two scenarios: Base and Scalability. In the Base Scenario, a certain load of hosted payloads was supported on the persistent platform. All of the value measures were given swing weights, but the most prioritized were the measures connected to the functions "sustain the hosted payloads" and "maintain communications." In the Scalability Scenario, the stakeholder wished to increase the hosted payload mass and power supported by the platform. This scenario prioritized the value measures connected to the functions "provide assembly of new payloads or support subsystems" and "provide manufacturing of support subsystems." In the next phase, Abstraction, the functions were connected to objectives and value measures.

Table 4.2: Morphological matrix, part I, for the baseline concept of operations.

Function	**Means**				
Launch resources (i.e. new support subsystems or manufacturing resources)	Dedicated CLV	Ride-along CLV	n/a		
Identify sufficient resources	Onboard imaging systems, image recognition software, and catalog on platform	Onboard imaging systems, image recognition software, and catalog on service module	n/a		
Deliver new subsystems from LV to platform	CLV upper stage	Service module that stays in LEO	New subsystems launched integrated with delivery subsystems (i.e. propulsion)	n/a	
Identify location to store resources	Onboard imaging systems, image recognition software, and catalog on platform	Onboard imaging systems, image recognition software, and catalog on service module	n/a		
Store resources	Module remains docked to platform and no unpacking necessary yet	Robotic arm on platform manipulates/stores resources	Robotic arm on service module	n/a	
Identify location of failed subsystem	Onboard imaging systems, image recognition software, and catalog on platform	Onboard imaging systems, image recognition software, and catalog on service module	n/a		
Remove failed subsystem and dispose of it	Robotic arm on platform	Robotic arm on service module	Explosive bolts	Active undocking and departure of the subsystem	n/a
Install new subsystem	Robotic arm on platform	Robotic arm on service module	Active docking of the subsystem	Completely replace module	n/a
Access resources for manufacturing	Robotic arm on platform	Robotic arm on service module	Conveyor belt	n/a	
Manufacture subsystem	Additive Manufacturing	n/a			

Table 4.3: Morphological matrix, part II, for the scalability scenario concept of operations.

Function	**Means**			
Launch platform modules	Dedicated CLV	Ride-along CLV		
Deliver platform modules to operational orbit	Each module has propulsion capability to transfer, and assembly happens later	Single module has propulsion capability to transfer, and assembly happens first	Service module for orbit transfer	
Assemble platform modules	Single module is passive hub, and other modules are active to dock	Single module is active hub and leads docking with each subsequent module	Every module is passive, and service module is active agent to lead docking of each module.	
Launch payloads	Dedicated CLV	Ride-along CLV		
Deliver payloads/subsystems from LV to platform	CLV upper stage	Service module for orbit transfer that stays in LEO	Payloads launched integrated with delivery subsystems (mainly propulsion)	n/a because payloads integrated on the ground
Perform proximity operations with platform	Platform is active during approach	Platform is passive during approach		
Perform rendezvous operations with platform	Platform is active for docking, and other module is passive	Platform is passive for docking, and other module is active		
Dock or berth to platform	Robotic arm on platform guides module with grappling points and coupling mechanism	Robotic arm on service module guides module with grappling points and coupling mechanism	Electromagnets on each module	Docking ports on all modules
Identify location to install payloads	Onboard imaging systems, image recognition software, and catalog on platform	Onboard imaging systems, image recognition software, and catalog on module	n/a because payloads integrated on the ground	
Install payloads	Payloads integrated with modules on the ground	Robotic arm on platform	Robotic arm on service module	
Sustain payloads, etc.	Subsystems on platform design for support…			
Maintain communications	Ka-band communications payload			
Maintain situational awareness	GPS and ADCS sensor suite			

4.2 Abstraction

The Abstraction Phase is used to frame the problem in more detail, including to identify classes of agents, agent interactions, and big-picture dynamics (DeLaurentis, 2005). This section will include discussion of:

- Stakeholders
- Stakeholder objectives, and how the objectives trace from functions and to value measures
- Drivers and disruptors
- Modeling of the actors (i.e., resources from the ROPE Table within the decision scope) and the network

Each of these items follows the process laid out in the MAPS workflow, starting at the beginning as seen in Fig. 4.1(b). The products of the Definition phase assisted the steps of the Abstraction phase. By the end of the Abstraction phase, all of the inputs to MAPS were created, including: design variables, value measures selection, scenarios, swing weights, a list of required and optional systems, and variable bounds. This set up the case study for the Implementation phase, where the tools in MAPS were applied.

4.2.1 Stakeholders

Stakeholders of a persistent platform venture may vary depending on if it is a private company or a government entity. Assuming the platform is developed by a private company, stakeholders would include: shareholders, company management, company employees, system operators, customers renting spots on the platform for their payloads, system manufacturers, and the science community in which the customers belong and interact (Parnell, 2016). Additionally, based on the ROPE table, stakeholders would include producers, sellers, and operators of launch vehicles; space

systems insurance companies; international governing bodies, e.g., UN COPUOS; and private investors. Stakeholders are implicit entities often not directly represented in the functional or physical architectures (DeLaurentis et al., 2019). However, the stakeholders have preferences which directly influence the objectives and value measures by which an architecture will be assessed. As displayed in the MAPS Analysis Framework (Fig. 3.8), the definition of stakeholders is followed by definition of stakeholder objectives in the decision opportunity.

4.2.2 Stakeholder Objectives in the Functional Value Hierarchy

A Functional Value Hierarchy (FVH) is a recommended method for generating objectives and tracing them to value measures (Parnell, 2016). This hierarchy represents the tiers:

$$\text{Primary Objective} \rightarrow \text{Functions} \rightarrow \text{Objectives} \rightarrow \text{Value Measures}$$

The FVH developed for the persistent platform concept is provided in Fig. 4.4. Some value measures were obvious, such as the total mass of hosted payloads and total power for hosted payloads. Others were added to reflect an interest in non-traditional value measures, such as platform complexity and several flexibility category measures. Others were a result of considering the operational scenarios, such as the presence of assembly or manufacturing capability, cost of upgrade, and complexity of upgrade.

There were some reasons that a persistent platform with OSAM would be considered valuable that didn't show up on the FVH. As discussed in literature (Williams et al., 2018), (Piskorz and Jones, 2018):

- OSAM can circumvent the volume restrictions of launch vehicles
- On-orbit manufacturing circumvents the need for "parasitic mass" added to a system to withstand the launch environment
- Smaller/lower-mass payloads reduce the reliance on heavy lift launch vehicles, opening the architecture up to a broader range of launch providers

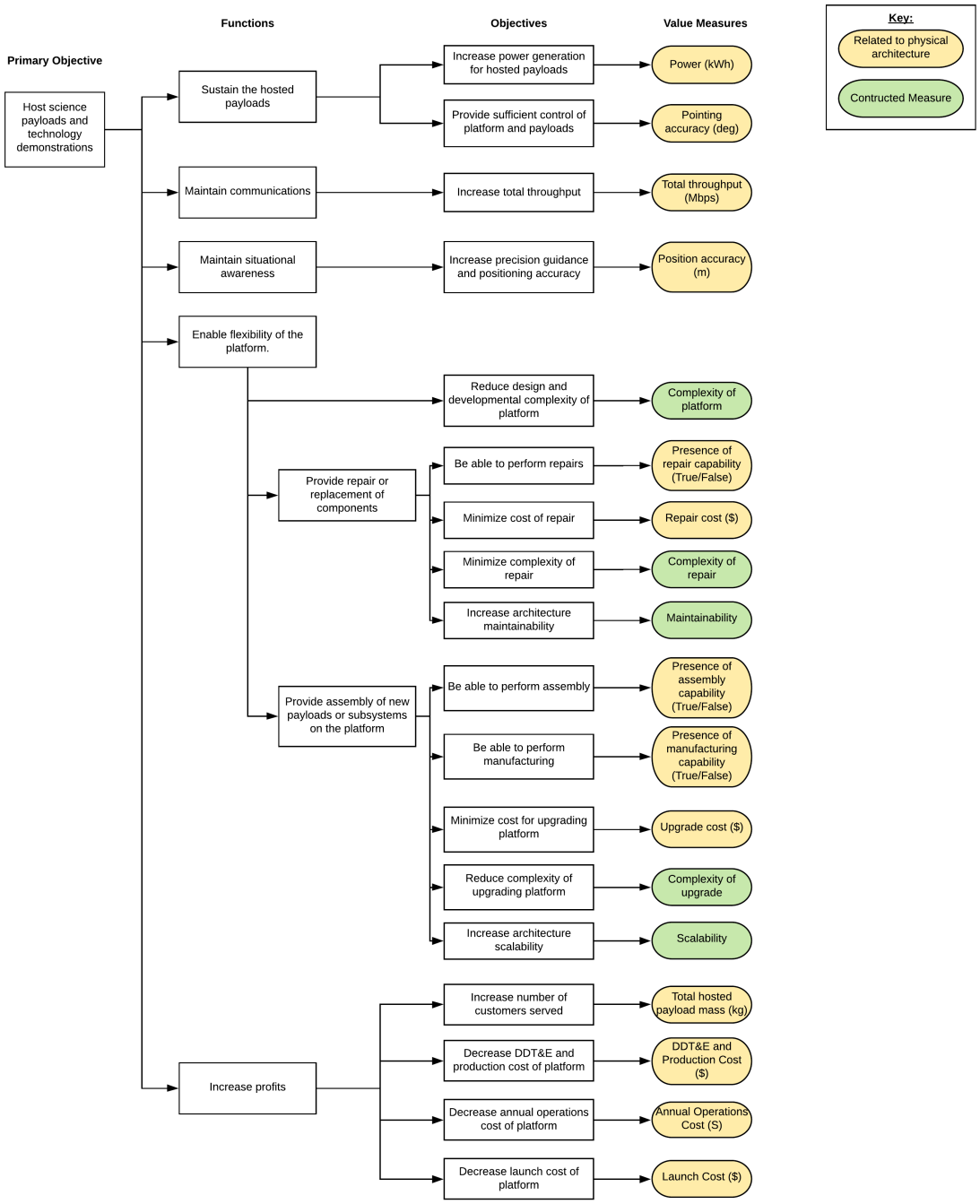


Figure 4.4: Functional Value Hierarchy created for the OSAM persistent platform architecture of this case study.

- Reduces reliance on large, expensive ground facilities used for production and integration
- Potential to spread the cost over multiple years, through modularity and incremental buildup
- Potential to spread the cost across multiple partners/programs

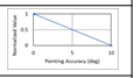
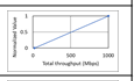
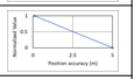
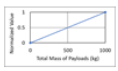
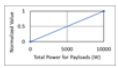
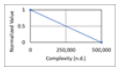
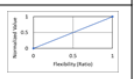
While other studies may have presented means of evaluating different measures not reflected in this FVH, the list presented here represents a sufficient diversity for demonstrating the capabilities of analysis in MAPS.

4.2.3 Evaluating the Value Measures

To evaluate additive value in MAPS, three items are needed: a value measures table, swing weights, and a tool to evaluate each measure. First, the value measures table is how the list of measures are input to MAPS. The table also includes value functions, by which the measures are normalized in the Additive Value Tool. The value functions are defined by the type—e.g., linear—plus the minimal acceptable value and ideal value, as shown in Tables 4.4-4.5. It is important that the minimal acceptable value and the ideal value are carefully selected for each value measure, so the measure can be appropriately normalized. The table also includes other information, including the type of measure (natural or constructed) and the source in MAPS where the evaluation would be performed (Parnell, 2016). For the case study, the table was input using an Excel spreadsheet, which MAPS could easily read.

Next, a set of *swing weights* was created. Swing weights are used to weight the value measures in the Additive Value Tool. A swing weight matrix was created for the Base Scenario following the guidelines of (Johnson et al., 2013). In the matrix, each value measure is placed into a column based on its importance and placed into a row based on the range of the measure. Next, *matrix weights* are assigned based on

Table 4.4: Value measures table, part I.

Function	Objective	Value Measure	Type	Data Source	Minimal Acceptable Value	Ideal Value	Value Function
Sustain the hosted payloads	Provide guidance, control, and navigation capabilities	Pointing accuracy (deg)	Natural	Design – Space Systems Design Tool	10.0	0.1	
Maintain communications	Increase total throughput	Total throughput (Mbps)	Natural	Design – Space Systems Design Tool	20	1,000	
Maintain situational awareness	Increase precision guidance and positioning accuracy	Position accuracy (m)	Natural	Design – Space Systems Design Tool	5	0.1	
Increase profits	Increase number of customers served	Total mass of hosted payloads (kg)	Natural	Design – Architecture Enumeration Tool	1	1,000	
		Total power for hosted payloads (W)	Natural	Design – Architecture Enumeration Tool	1	10,000	
Enable flexibility of the platform	Reduce design and developmental complexity of the platform	Complexity of platform (n.d.)	Constructed	Complexity Tool	500,000	0	
	Increase architecture adaptability	Adaptability	Constructed	Flexibility Tool	0	1	

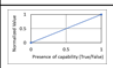
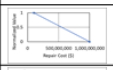
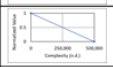
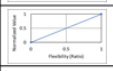
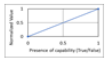
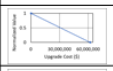
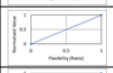
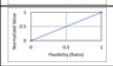
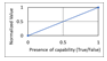
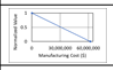
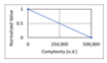
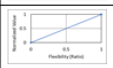
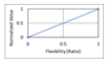
the location in the matrix. [1] Swing weights are then calculated based on the matrix weights such that they sum to equal one (Johnson et al., 2013). The swing weights for the Base Scenario are shown in Table 4.6.

A separate set of swing weights was created for the Scalability Scenario, as seen in Table 4.7. The operational scenario provided an opportunity to place a higher priority on certain measures, including: presence of assembly capability, scalability, position accuracy, complexity of platform, and complexity of upgrade. The matrix weights and therefore the swing weights were updated accordingly. The sets of swing weights are also included in the Excel spreadsheet that is input to MAPS.

The final step was the actual evaluation of each measure using a tool in MAPS. The specific tool or data source was already identified in the value measures table, so the user next needed to actually create a script in MAPS that called each tool or data source. The full value measures data was recorded for each architecture alternative in

[1] A detailed view of how the matrix weights were progressively assigned based on the value measure's importance and range can be found in Appendix Fig. C.1.

Table 4.5: Value measures table, part II.

Function	Sub-Function	Objective	Value Measure	Type	Data Source	Minimal Acceptable Value	Ideal Value	Value Function
Enable flexibility of the platform *(contd.)*	Provide repair or replacement of components	Be able to perform repairs	Presence of repair capability (True/False)	Natural	Architecture Enumeration Tool	0	1	
		Minimize cost of repair	Repair cost ($)	Constructed	Life Cycle Cost Tool	1×10^9	1×10^8	
		Minimize complexity of repair	Complexity of repair (n.d.)	Constructed	Complexity Tool	500,000	0	
		Increase architecture maintainability	Maintainability	Constructed	Flexibility Tool	0	1	
	Provide assembly of new payloads on the platform	Be able to perform assembly	Presence of assembly capability (True/False)	Natural	Architecture Enumeration Tool	0	1	
		Minimize cost for upgrading platform	Upgrade cost ($)	Constructed	Life Cycle Cost Tool	1×10^9	1×10^8	
		Increase architecture scalability	Scalability	Constructed	Flexibility Tool	0	1	
		Increase architecture evolvability	Evolvability	Constructed	Flexibility Tool	0	1	
	Provide manufacturing of new payloads	Be able to perform manufacturing	Presence of manufacturing capability (True/False)	Natural	Architecture Enumeration Tool	0	1	
		Minimize cost to manufacture payloads	Manufacturing cost ($)	Constructed	Life Cycle Cost Tool	1×10^9	1×10^8	
		Reduce complexity of manufacturing	Complexity of manufacturing (n.d.)	Constructed	Complexity Tool	500,000	0	
		Increase architecture scalability through manufacturing	Scalability	Constructed	Flexibility Tool	0	1	
		Increase architecture evolvability through manufacturing	Evolvability	Constructed	Flexibility Tool	0	1	

MAPS and was passed into the MAPS Additive Value Tool together with the swing weights.

4.2.4 Drivers and Disruptors

The next product of the Abstraction phase is identification of drivers and disruptors of the SoS. DeLaurentis et al. define drivers as "those things that determine the forcing functions that drive the stakeholder network and receive a feedback from the stakeholders", are external to the SoS, and do not physically affect the consumer

Table 4.6: Swing weights for the Base Scenario.

Impact of the Range of the VM	Mission Critical			Enabling			Enhancing		
	Value Measure	Matrix Weight	Swing Weight	Value Measure	Matrix Weight	Swing Weight	Value Measure	Matrix Weight	Swing Weight
Significant Impact	Total mass of hosted payloads	100	0.106	Presence of assembly capability	85	0.090	Adaptability	60	0.063
							Maintainability	60	0.063
	Total power for hosted payloads	100	0.106				Scalability	60	0.063
							Evolvability	60	0.063
Medium Impact	Total throughput	90	0.095	Presence of repair capability	70	0.074	Position accuracy	40	0.042
Minimal Impact	Pointing accuracy	80	0.085	Presence of manufacturing capabilities	50	0.053	Complexity of platform	30	0.032
							Complexity of repairs	20	0.021
							Complexity of upgrade	20	0.021
							Repair cost	10	0.011
							Upgrade cost	10	0.011

Table 4.7: Swing weights for the Scalability Scenario.

Impact of the Range of the VM	Mission Critical			Enabling			Enhancing		
	Value Measure	Matrix Weight	Swing Weight	Value Measure	Matrix Weight	Swing Weight	Value Measure	Matrix Weight	Swing Weight
Significant Impact	Total mass of hosted payloads	100	0.100	Scalability	80	0.080	Adaptability	55	0.053
	Total power for hosted payloads	100	0.100				Maintainability	55	0.053
	Presence of assembly capability	95	0.095				Evolvability	55	0.053
Medium Impact	Total throughput	90	0.090	Presence of repair capability	65	0.065	Complexity of repairs	20	0.019
Minimal Impact	Pointing accuracy	75	0.075	Presence of manufacturing capabilities	50	0.050	Repair cost	10	0.010
	Position accuracy	70	0.070	Complexity of platform	40	0.040	Upgrade cost	10	0.010
				Complexity of upgrade	30	0.030			

(DeLaurentis, 2005), (DeLaurentis et al., 2019). In this case study, key drivers are the launch market and national space exploration goals.

The state of the global launch market may drive feasibility of a persistent platform concept. If launch prices are dramatically reduced and launch vehicles become significantly more available, there might not be a business case for a persistent platform. In this case, the barrier to space has decreased, and users may accept the cost to build or procure a full satellite bus for their payload(s) instead of renting space on a persistent platform. In this case also, the value case for demonstrating OSAM technologies in LEO may be diminished, further weakening the case for a persistent OSAM platform in LEO. The driving force of the launch market makes it even more important to understand the status quo in this market, as was explored in the Definition Phase.

National space exploration goals may also drive feasibility of a persistent platform concept. Goals that involve large-scale or sustainable exploration missions can provide the basis of a value case for many OSAM capabilities. Then also, public/private partnerships are known to be an effective means to develop new and disruptive technologies and to kick-start new markets. A persistent platform would be a useful location to perform technology demonstrations before new exploration missions. Based on what national space exploration goals are, a new platform venture may find willing investors that increase the feasibility of the project.

Disruptors are entities that can cause the delay or cancellation of activates in the SoS and are external to the system, but unlike drivers, they are physically felt by the consumers (DeLaurentis et al., 2019). Disrputors of a persistent platform in LEO may include: space weather, launch failures, mechanical failures, or technology development delays. Space weather that causes radiation damage or strikes from orbital debris could cause system failures or temporary communications outages. Launch failures or delays, mechanical or computer failures, and technology development delays, e.g for new proximity operations systems or robotic manipulators, could also disrupt the SoS. Such disruptors cannot be predicted except with probabilistic models, so the best way to proceed in light of the disruptors is to design a concept that can be robust to any delays or failures.

4.2.5 Modeling the Agents

The final step of the Abstraction phase is to finally model the agents and interactions. In MAPS, the Space Systems Design Tool enables much of this step. This step also involved modeling the input design variables and bounds. It is the design variables that distinguish between alternative designs for the persistent platform concept.

The first design variable is total hosted payload mass. This is in fact a simple input to the MAPS Space Systems Design Tool. A payload object requires three input parameters: mass, power requirement, and control accuracy requirement. For this study, a hosted payload was instantiated by passing in these three parameters. The hosted payload mass was varied in MAPS within bounds from 100 to 1,000 kg to generate sets of architecture design alternatives. The other two were fixed parameters at 1,000 W total power requirement and 1°control accuracy requirement.

The other two design variables are the presence of servicing capability and manufacturing capability, so new OSAM payloads were defined for this case study. A three-step process was developed based on the morphological matrix (Tables 4.2-4.3). It must be understood that a complete design option can be defined from a morphological matrix by combining one means (in the columns) for each function (in the rows) (Cross and Roy, 2000). So the steps to define the payloads were as follows:

1. The functions in the morphological chart that were related to each category of assembly, servicing, and manufacturing were sorted.

2. A set of means combinations was put together for each category. Each means combination is one option.

3. Each combination was translated into the specific technologies needed for the means.

For example, Table 4.8 displays the servicing functions from Step 1 and the combinations of means from Step 2. Table 4.9 contains all the combinations—each of which is

an option—for assembly, servicing, and manufacturing. The full results of these three steps for each of the OSAM categories are provided in Appendix D, Tables D.1-D.9.

Table 4.8: Based on the morphological matrix, functions that fall within the servicing category and combinations of means to form three servicing options.

Category	Function	Option 1 – None	Option 2 – Payload or Module Install Only (Platform)	Option 3 – Payload Install and Subsystem Swap (Platform)
Servicing	Identify location to install payloads	n/a – payloads integrated on the ground	Onboard catalog, image recognition software, and imaging systems on platform	Onboard imaging systems, image recognition software, and catalog on platform
	Install payloads	n/a – payloads integrated on the ground	Robotic arm on platform (RAI)	Robotic arm on platform (RAS)
	Identify sufficient resources	n/a – no maintenance capability	n/a – no maintenance capability	Onboard imaging systems, image recognition software, and catalog on platform
	Identify location to store resources	n/a – no maintenance capability	n/a – no maintenance capability	Onboard imaging systems, image recognition software, and catalog on platform
	Store resources	n/a – no maintenance capability	n/a – no maintenance capability	Module remains docked to platform and no unpacking necessary yet
	Identify location of failed subsystem	n/a – no maintenance capability	n/a – no maintenance capability	Onboard imaging systems, image recognition software, and catalog on platform
	Remove failed subsystem and dispose of it	n/a – no maintenance capability	n/a – no maintenance capability	Robotic arm on platform (RAS)
	Install new subsystem	n/a – no maintenance capability	n/a – no maintenance capability	Robotic arm on platform (RAS)

Table 4.9: List of all options for OSAM capabilities, which are design variables. Each option is mapped to the payloads required, using the payload IDs from Table 4.10.

Category					
Assembly	Option Names:	Option 1 – Passive platform modules during docking with assist from Service Module			
	Payload IDs for each Option:	1			
Servicing	Option Names:	Option 1 – None	Option 2 – Payload or Module Install Only (Platform)	Option 3 – Payload Install and Subsystem Swap (Platform)	
	Payload IDs for each Option:	None	2, 4	3, 4	
Manufacturing	Option Names:	Option 1 – None	Option 2 – Manufacture Platform Structures	Option 3 – Manufacture Structures + Solar Arrays	Option 4 – Manufacture Structures + Solar Arrays + Electronics
	Payload IDs for each Option	None	3, 5	3, 6	3, 7

The end result was a list of seven OSAM payloads, displayed in Table 4.10. The payloads are mapped to one option for assembly, three options for servicing, and four options for manufacturing in Table 4.9. There was only one payload combination for assembly since from the fixed parameters, all architectures had the same assembly capability. The servicing and manufacturing options are the options for the two design variables.

As displayed in Table 4.10, each OSAM payload was defined by the three payload parameters required by the Space Systems Design Tool—mass, power requirement, pointing accuracy requirement. Additionally, each was defined by their instrument cost category for the NICM and an adjacency matrix, which was used as input to the Complexity and Flexibility Tools. All details on the payloads are also provided in Appendix D (starting with Fig. D.1). All the data generated for the OSAM options and payloads was added to the Excel spreadsheet that was input to MAPS.

Modelling the actors and interconnections for the Abstraction phase concludes with a depiction of the physical architecture. Both the functional architecture and design variables were necessary to fully picture the physical architecture. One notional configuration is provided in Fig. 4.5. For another perspective of the physical systems, graphics on the concept of operations were created, as seen in Figs. 4.6-4.7. The interconnectivity of agents is an important part of analysis in MAPS. The physical architecture shows the connections between the assemblies in the concept, and within those the connections between modules, and within those the connections between subsystems. During the Implementation phase, next, the impact of the interconnections and the impact of altering the configurations was evaluated.

Table 4.10: OSAM payloads created to serve different options for servicing, assembly, and manufacturing.

Payload	Payload ID	Properties					
		Dry Mass (kg)	Wet Mass (kg)	Power Req. (W)	Pointing Accuracy Req. (deg)	Position Accuracy (m)	NICM Instrument Category
Assembly Package (AP) - 2x SPA, 2x SPP	1	13.45	13.45	30	0.15	0.1	1
Robotic Arm for Installation (RAI + SPA)	2	83.15	83.15	145	0.15	0.1	1
Robotic Arm for Subsystem Servicing (RAS + SPA)	3	108.15	108.15	188	0.15	0.01	1
Onboard imaging systems, image recognition software, and catalog (OIS)	4	6.5	6.5	27.5	0.1	nan	1
Robotic Arm for Subsystem Servicing (RAS + SPA)	3	108.15	108.15	188	0.15	0.01	1
Manufacturing Package 2 (SPA + LVV + AMM + TFS)	5	72.65	72.65	320	nan	0.01	1
Manufacturing Package 3 (SPA + LVV + AMM + TFS + FBS)	6	92.65	92.65	320	nan	0.01	1
Manufacturing Package 4 (SPA + LVV + AMM + TFS + FBS)	7	102.65	102.65	320	nan	0.01	1

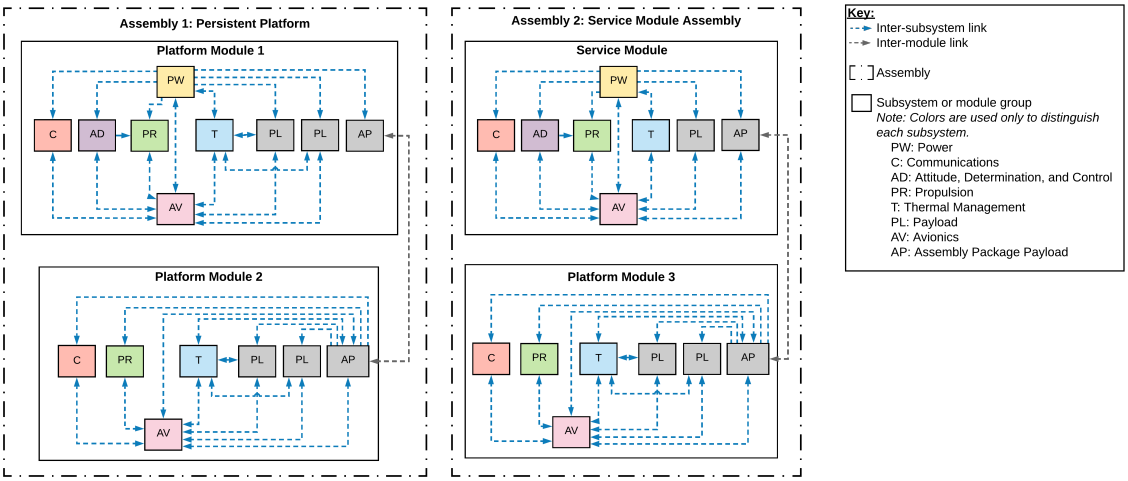


Figure 4.5: Paper model for an example persistent platform architecture. Each platform module has an Assembly Package (AP) payload and forms an assembly in orbit, while the Service Module docks to a subsequent platform module to transfer it to the platform.

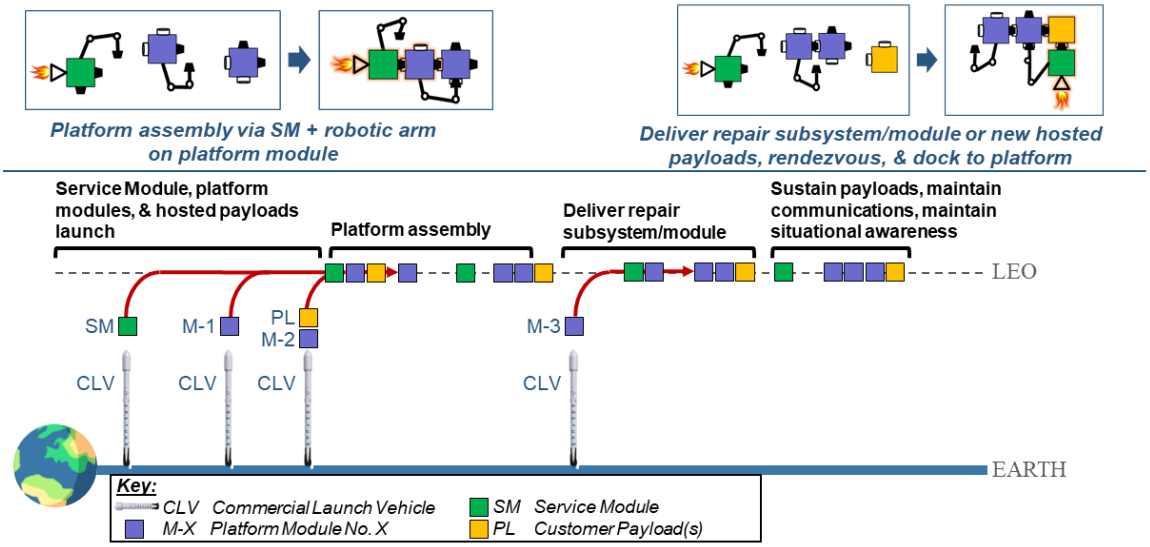


Figure 4.6: Concept of operations graphic including the platform concept's physical systems.

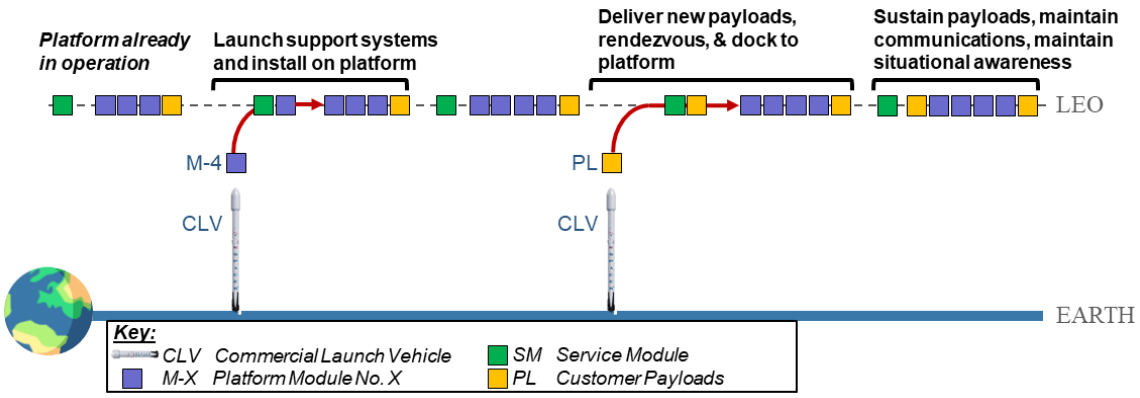


Figure 4.7: Concept of operations graphic for scalability scenario.

4.3 Implementation

The final phase of SoSE modeling and analysis is Implementation, where the model or simulation is applied, results are collected, and trade-offs are interpreted (DeLaurentis, 2005). These are also the remaining steps in the decision management process described by the SEBoK, namely to generate creative alternatives, assess alternatives, communicate trade-offs, etc. (see Fig. 2.2). This section will discuss

the cases run in MAPS, how the models were created, and the value analysis for the architecture alternatives.

4.3.1 Run Cases of Design Variables

The total number of possible combinations of the three design variables is quite large, especially with a continuous variable. For a reasonable amount of data, three options were selected for total hosted payload mass: 100, 500, and 1,000 kg. Then, there were three servicing options and four manufacturing options. In total, thirty-six cases were run, where each case represented one combination of design variables. Further, each design option was passed to the MAPS Architecture Enumeration Tool. Enumerated architecture alternatives represented the number of ways that the payloads/fractionated subsystems could be shuffled between a number of modules. This resulted in a total of 18,876 different architecture alternatives.

Three additional cases were also run. Three "baseline," or "no OSAM," architectures were defined, each with 100, 500, or 1,000 kg of hosted payload mass and no OSAM payloads at all. The "no OSAM" architectures were useful as comparisons to the design cases where OSAM capabilities are added. In comparison, even architectures in the other cases with no servicing or manufacturing—defined by the design variables—still had assembly, since this was a fixed parameter (discussed in Section 4.1.1 during the Definition Phase). The "no OSAM" architectures were still evaluated for the repair and scalability operations, although they required alternate strategies to accomplish the operations since in-orbit assembly, servicing, and manufacturing were not present.

4.3.2 How the Models were Created in MAPS

In the MAPS workflow, the Implementation phase is the point at which tools were applied directly in the MAPS coding environment, as highlighted in Fig. 4.1(c). At a high level, this phase required three new files to be created: an input Excel

spreadsheet, a runner script, and a data analysis script. An overview of each of these files is given in this section.

The Excel spreadsheet was created during the Abstraction phase. It contained the list of value measures and value functions, swing weights, OSAM options, and payload definitions. A new script was created for this case study to read the Excel file; however, future studies may be able to use this same function, reducing the workload required from new users.

The second important new file was the "runner" script. The runner was created to:

1. Record the design variable inputs
2. Create the list of fractionated subsystems
3. Call the Architecture Enumeration Tool
4. Call the Space Systems Design Tool
5. Call the value evaluation script
6. Save results in output files

The runner will of course call other files in MAPS as necessary, but it is the key interface to run each case.

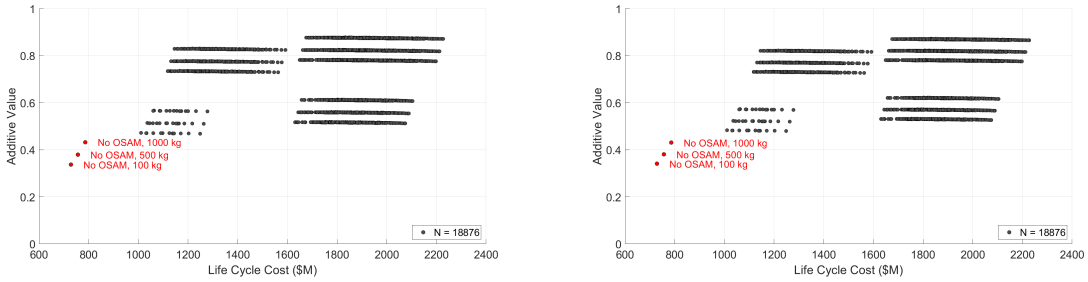
The list of fractionated subsystems is the only input to the Architecture Enumeration Tool. The fractionated subsystems for this case study included the hosted payload, the selected OSAM payloads, and two specific subsystems related to the platform design strategy—power and ADCS. The purpose is to enumerate how many ways the payloads/subsystems can be shuffled between a number of modules. It must therefore be understood that the fractionated subsystems are required in the architecture but only optionally present on individual modules. For example, the single hosted payload object is required in the architecture, but it may be integrated with any one of the modules in the architecture. Changing the number of modules and

which fractionated subsystems are on which module impacts the launch cost and cadence and the interconnections in adjacency matrices, which impacts measures of complexity and flexibility.

The last key new file is the data analysis script. This script was created for this case study to load the saved data, create graphs, and identify a small number of interesting architectures. The results analyzed with this file will be discussed in the following section.

4.3.3 Architecture Value Analysis

The full tradespace is graphed for additive value versus life cycle cost in Fig. 4.8a for the Base Scenario and in Fig. 4.8b for the Scalability Scenario. The number of architecture alternatives is included in the figures as indicated by "N=18876" in the bottom right. The difference in additive value between the Base and Scalability scenarios is the result of different swing weights. However, since only a few value measures in this study changed in level of priority, and since there were so many value measures contributing to additive value, it's observed that the results do not change significantly between scenarios. Perhaps individual architectures stand out between the scenarios, so a deeper look at the data is required.



(a) Base Scenario (b) Scalability Scenario

Figure 4.8: Comparison of full tradespace results for additive value vs. life cycle cost between the Base and Scalability Scenarios. Changes to the swing weights for the Scalability Scenario did not result in any significant differences to additive value.

The cause of several interesting effects observed in the full tradespace are only revealed when using coloring or marker sizes. Figure 4.9 shows the full tradespace colored by the servicing option and also uses marker sizes based on the total hosted payload mass. Similarly, Fig. 4.10 shows the tradespace colored by the manufacturing option. The following graphs all present data from the Base Scenario only.

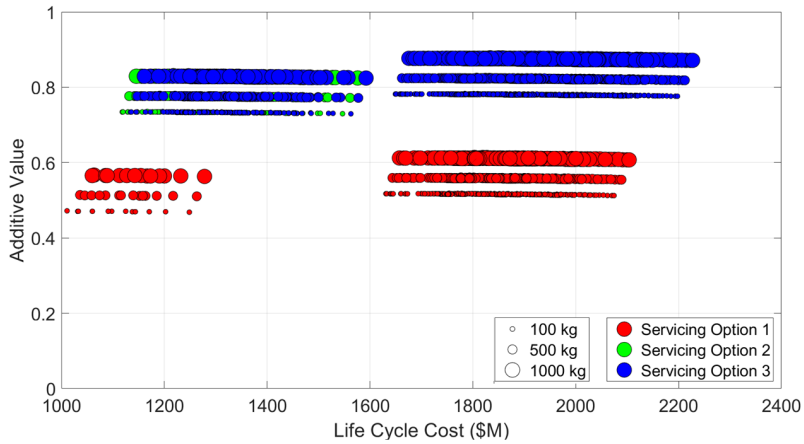


Figure 4.9: Full tradespace colored by the servicing option and with marker sizes to indicate total mass of hosted customer payloads. Addition of servicing capability and increases in hosted mass correspond to increases in value at limited additional cost.

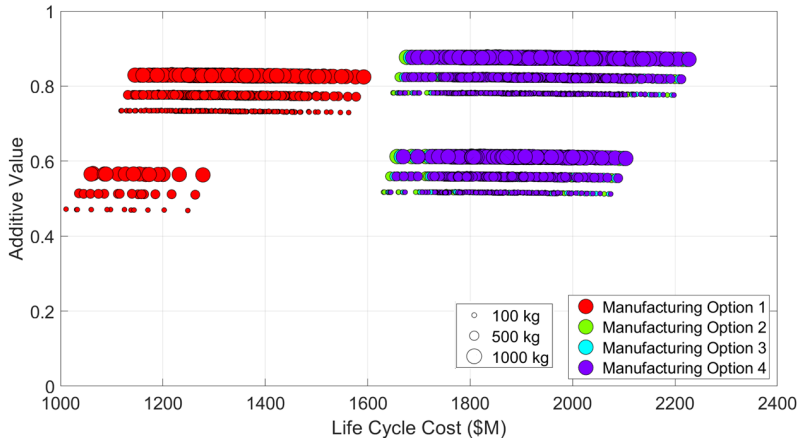


Figure 4.10: Full tradespace colored by the manufacturing option and with marker sizes to indicate total mass of hosted customer payloads. Addition of manufacturing capability adds minimal value for large increases in cost.

The first effect is that the data falls into subsets of nearly horizontal lines. This is caused by small variations in enumerated architectures. When architecture alternatives were enumerated for a given design option, changing the number of modules caused only slight variations in some of the value measures. For example, there would have been a relatively small difference in complexity between one architecture with two modules and another architecture with three modules. Additionally, the complexity value was then normalized in the Additive Value Tool, further reducing its impact on the final additive value. However, increasing the number of modules certainly increased the life cycle cost. Added launch costs and development costs for each additional module contributed to a horizontal spread. Thus, nearly horizontal lines of data resulted for each design option.

The second effect is significant spikes in value between sets of horizontal lines. This is surely due to a change in a design characteristic that had high priority (i.e., a high swing weight) in the additive value model. In Figs. 4.9-4.10, it's observed that an increase in hosted payload mass caused a significant increase in additive value, on the order of 10% or 20% when increasing from 100 to 500 kg or from 100 to 1,000 kg, respectively. This effect was expected because the highest priority was given to the mass of customer payloads hosted on the persistent platform. However, the increase in mass did not require a significant increase in life cycle cost.

The presence of servicing capability and manufacturing capability also caused spikes in value and cost. From Fig. 4.9, it's observed that the addition of servicing capability added significant value, on the order of 50%. The addition of servicing capability required a cost increase on the order of $100 million, or approximately a 10% increase to an architecture with a life cycle cost on the order of $1,000 million. However, the distinction between Servicing Options 2 and 3 did not appear to impact the additive value. As a reminder, Servicing Option 1 meant no servicing capability (see Table 4.8). The difference between Servicing Options 2 and 3 was the capability to do module installation versus the capability to perform specific subsystem swaps.

Therefore, it can be concluded that any level of servicing capability could be added to an architecture to improve additive value at minimal additional total cost.

In comparison, the addition of manufacturing capability provided a smaller increase to additive value, on the order of 5%, but the cost increase was on the order of $500 million, or approximately 50%. Also, the increasing level of manufacturing onboard the platform did not seem to matter in terms of cost or value, since the results for manufacturing options 2 vs. 3 vs. 4 were overlapping in Fig. 4.10. The Manufacturing Option 1 represented no manufacturing capability, whereas the Options 2, 3, and 4 represented increasing capability. In Option 2, the platform could manufacture structures; in Option 3, structures and solar arrays; and in Option 4, structures, solar arrays, and electronics. It is observed that adding manufacturing capability did not ultimately reduce the life cycle cost nor increase the additive value.

Figure 4.11 shows the full tradespace colored by the number of modules. This reflects the design characteristic of fractionating the subsystems and payloads between a number of free-flying modules. The modules are assembled in orbit to form the persistent platform. In every case, increasing the number of modules (above two) adds significant cost while at the same time slightly reducing the additive value. The increase in cost with the increase in the number of modules that is observed in Fig. 4.11 can be explained by the additional launch and development costs. The reduction in additive value might be explained by a few factors. Architectures with a higher number of modules likely had higher complexity of the platform. Lower complexity was preferred, so this would have reduced the additive value. For other factors, a deeper look at the individual value measure evaluations is required to distinguish architectures.

The one-module architecture won for lowest cost and highest value only for architectures with no servicing or manufacturing payloads. As soon as new OSAM payloads were added, several two-module architectures scored better than one-module architectures. This effect is likely due to the increased cost and complexity required to replace an entire one-module platform that had expensive OSAM payloads on-

board. Comparatively lower repair cost and/or repair complexity might have led to higher additive value.

Finally, the baseline architectures stand apart from the rest of the data. The baseline architectures each had only a one module and no OSAM payloads at all. It is easy to understand that their overall value was much lower. In comparison, the subset of architectures in the bottom left corner of the tradespace had an assembly package, which increased both the additive value and cost. The baseline architectures are valuable for understanding the ultimate added value by including certain OSAM capabilities.

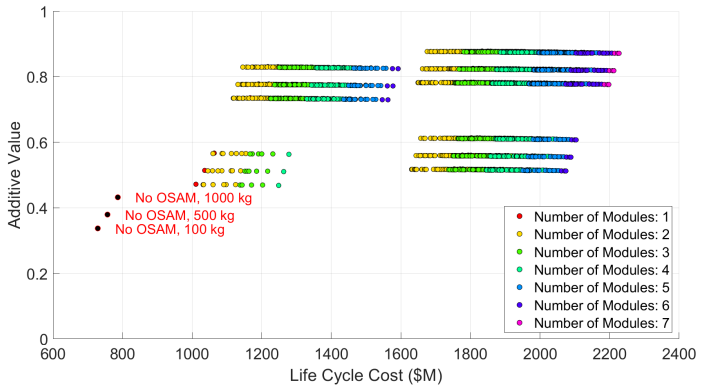


Figure 4.11: Full tradespace colored by the the number of modules in the persistent platform. Addition of modules causes small reductions in value and increases life cycle cost—primarily through increased development and launch costs.

A selection of architectures along the Pareto frontier is highlighted in Fig. 4.12. A Pareto frontier represents the architectures that have the maximum value at each level of cost, indicating that they have optimized the design variables. The other architectures are deterministically dominated by the ones on the frontier; these would not be selected, since there are better options with higher value and lower cost.[2] Future work might update this Pareto front by running additional cases, for example

[2]It should be noted that this analysis was purely deterministic and did not consider uncertainty. In the decision management process, eliminating architectures without considering uncertainty is not recommended (Parnell, 2016).

outside the current bounds on hosted payload mass or with different combinations of OSAM capabilities.

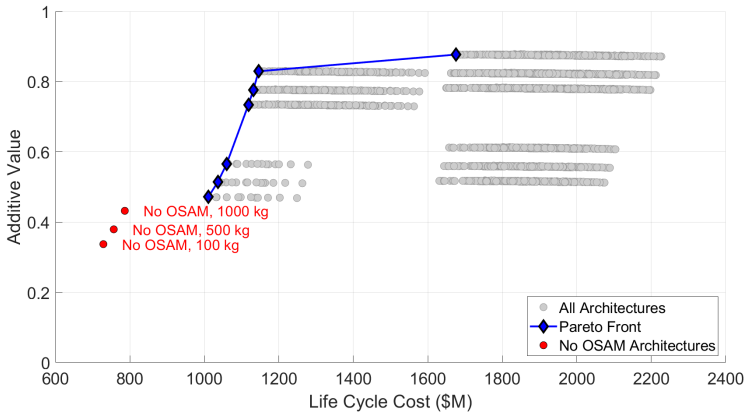


Figure 4.12: Architectures are identified on the Pareto front from the full tradespace, representing the optimal balance of value and cost.

The final perspective of the full tradespace compares a different performance measure. Since some analysts find it valuable to compare architectures based on total mass delivered to orbit, Fig. 4.13 shows the results for total mass versus life cycle cost. It was expected that an increase in total mass would correspond with an increase in life cycle cost, since mass is one parameter which impacts cost. However, this graph does not tell a complete story for decision-making. Compared to Fig. 4.8, there is clearly a dramatic impact when considering a collection of value measures that have been specifically selected to reflect stakeholder objectives.

Figure 4.14 displays a selection of four architectures from the Pareto front and a table providing more information on each. The first four rows of the table display design characteristics, and cells have been highlighted yellow where the architecture has interesting options that set it apart from the rest, e.g., it does have some level of OSAM capability or it has more than one module. The final six rows display the results of analysis and are highlighted in green. This is one means to visualize further details of the results or to narrow in on the architecture alternatives that provide the highest overall value for a given cost.

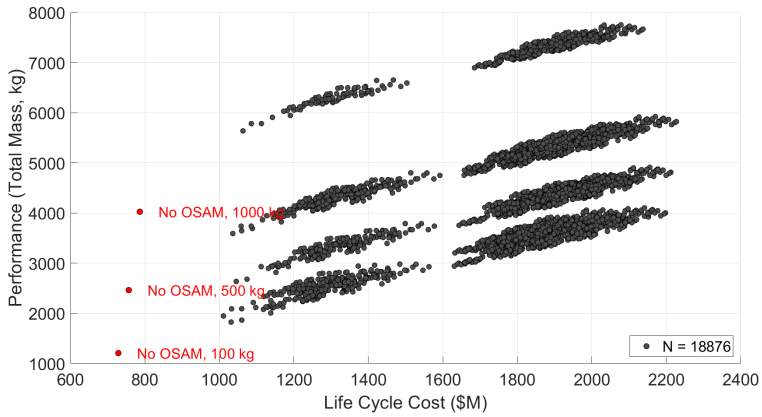
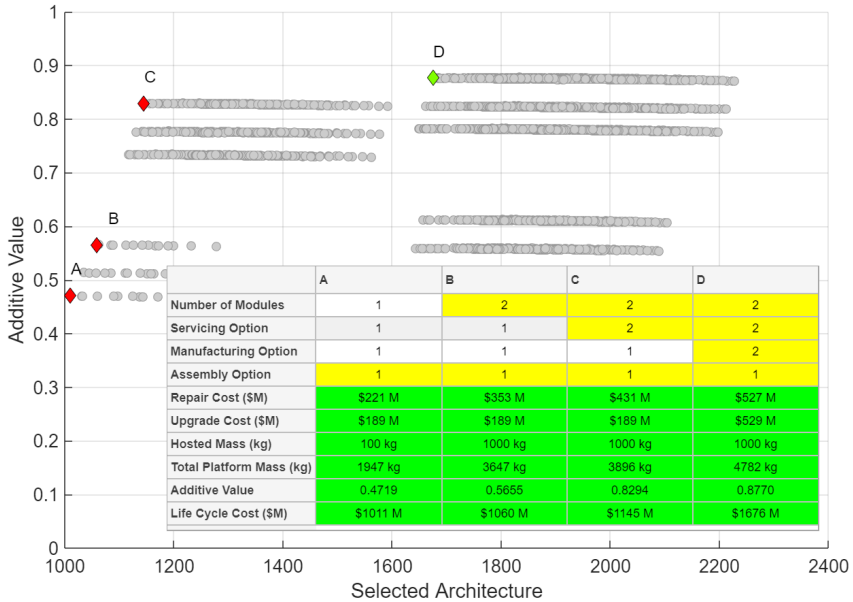


Figure 4.13: Total mass in orbit is treated as a performance measure and is graphed vs. life cycle cost for the full tradespace. Increase in mass and cost are correlated, but this graph is not able to provide a sufficient view of the trade-offs for overall value.



	A	B	C	D
Number of Modules	1	2	2	2
Servicing Option	1	1	2	2
Manufacturing Option	1	1	1	2
Assembly Option	1	1	1	1
Repair Cost ($M)	$221 M	$353 M	$431 M	$527 M
Upgrade Cost ($M)	$189 M	$189 M	$189 M	$529 M
Hosted Mass (kg)	100 kg	1000 kg	1000 kg	1000 kg
Total Platform Mass (kg)	1947 kg	3647 kg	3896 kg	4782 kg
Additive Value	0.4719	0.5655	0.8294	0.8770
Life Cycle Cost ($M)	$1011 M	$1060 M	$1145 M	$1676 M

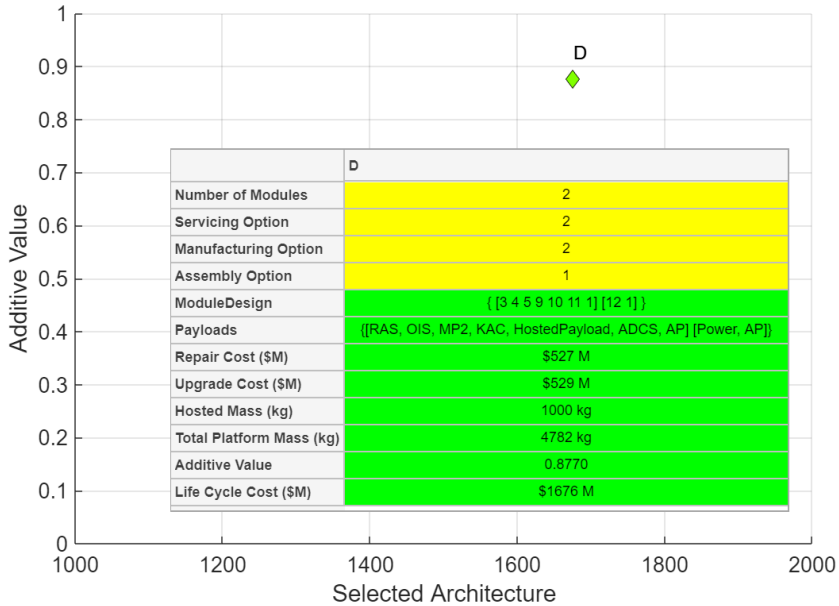
Figure 4.14: Four Pareto front architectures were selected for further study. A table displays design characteristics in the first four rows, where interesting characteristics like two modules, servicing option 2, and manufacturing option 2 are highlighted in yellow. The final six rows are results of analysis, highlighted in green.

Additional design characteristics and value measure results could be explored in other variations of the table. Figure 4.15 illustrates an example point design and a

table of design characteristics. Certain observations can be made when reviewing the repair and upgrade costs in the context of the module design.

First, the the repair operations are considered, wherein the power subsystem was required to be replaced. It is observed for this point design that out of the total $\$1,676$ million life cycle cost, \$527 million is required for the repair cost. Reviewing this architecture's module design, it is observed that the platform had a Robotic Arm for Subsystem Swap (RAS), which means it could replace a single subsystem rather than a full module (as described in Table 4.10). With an RAS, it would make sense that this architecture had a lower repair cost compared to other architectures that had to replace a full module and possibly lower life cycle cost as a result, which is likely why this architecture is on the Pareto front. Also, the power subsystem was in a module separate from the other payloads. With less direct linkages from the power subsystem to payloads and subsystems to remove and replace during the repair operations, it would also make sense that the upgrade complexity was lower compared to other architectures. This may also have given it a slight value boost.

Next, the scalability operations are considered, wherein a stakeholder defined that the support capability should increase by 50 kg of payload mass and 100 W for payload power. From Fig. 4.15, \$529 million in upgrade cost is required for this architecture. Reviewing this architecture's module design, it is observed that the platform had the Manufacturing Package 2 (MP2) onboard, which means it could manufacture additional structures (as described in Table 4.10). MP2 would have been used to manufacture structures needed to support additional hosted payload mass or power, but the remaining support subsystems and payloads would need to be delivered to the platform. All of these would have been installed by the RAS onboard the platform. The upgrade cost and complexity includes the costs and complexities of MP2 and RAS as well as the costs and complexities of the newly-delivered support subsystems. Observing the additional design characteristics in this way therefore reveals a lot of information about this Pareto front architecture.



	D
Number of Modules	2
Servicing Option	2
Manufacturing Option	2
Assembly Option	1
ModuleDesign	{ [3 4 5 9 10 11 1] [12 1] }
Payloads	{[RAS, OIS, MP2, KAC, HostedPayload, ADCS, AP] [Power, AP]}
Repair Cost ($M)	$527 M
Upgrade Cost ($M)	$529 M
Hosted Mass (kg)	1000 kg
Total Platform Mass (kg)	4782 kg
Additive Value	0.8770
Life Cycle Cost ($M)	$1676 M

Figure 4.15: One example point design and a table where the first four rows are design characteristics (highlighted in yellow) and the next rows are results of analysis (highlighted in green).

Figure 4.16 displays the normalized, weighted value contributions for the full list of seventeen value measures. (The values are normalized and weighted in the additive value model, as explained in Section 2.2.2, Eqs. 2.1-2.2.) The chart compares the four selected Pareto architectures against the "no OSAM" architectures for 100 kg and 1,000 kg hosted payload mass, the hypothetical best, and the ideal. In comparison to the "no OSAM" architecture for 100 kg, architecture A scores better for two value measures: position accuracy and presence of assembly capability. In comparison to the "no OSAM" architecture for 1,000 kg, architectures B, C, and D scored better for several value measures, such as: presence of assembly, repair, and/or manufacturing capabilities, position accuracy, and scalability. However, both complexity of the platform and complexity of repairs are higher in architectures B, C, and D than in the "no OSAM" architecture. Architecture D has a higher complexity of upgrade due to the presence of manufacturing payloads. The stacked bar chart is a helpful visualization technique for the full list of value measures.

In Fig. 4.16, the "ideal" bar represents the maximum scores possible, and the "hypothetical best" represents the best scores actually achieved across all architectures in the tradespace (Parnell, 2016). The difference between the ideal—by definition in the additive value model, 1.0—and a given architecture's additive value represents the value that has not been achieved by that architecture. The difference between the hypothetical best—in this case study, 0.8856—and the ideal additive value represents the technology gap, where the technologies in this study are not capable of achieving the full ideal value (Parnell, 2016). This means that the OSAM technologies and hosted payload support in this study are still not enough to obtain maximum value, with a gap of 0.1144.

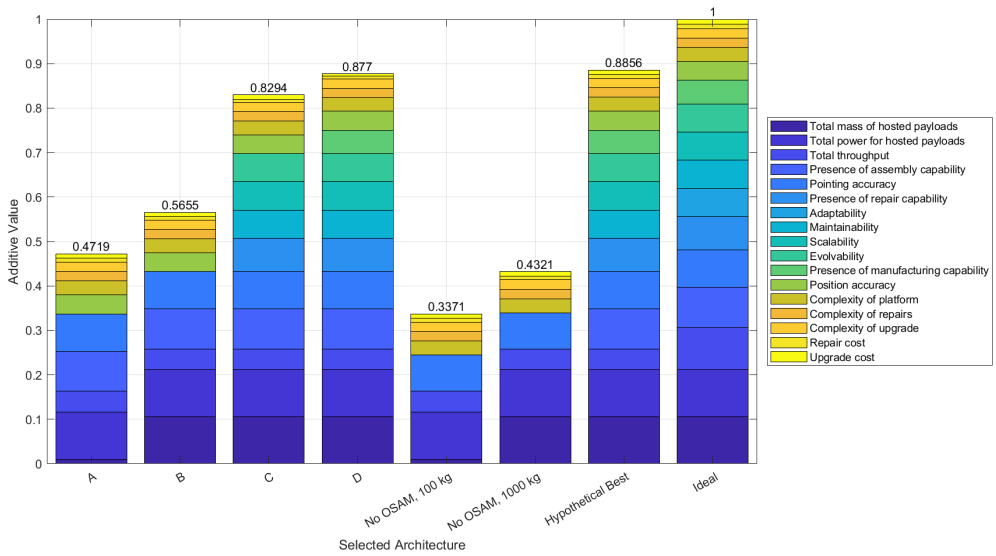


Figure 4.16: Normalized, weighted contributions from all 17 value measures are stacked. The gap between the Pareto architectures and the hypothetical best represents the value that could feasibly be achieved if the architecture was improved. The gap between the hypothetical best and the ideal represents the limit of the modeled technologies in achieving full value for stakeholders.

The last data analysis technique used in this case study was the creation of a spider plot. Figure 4.17 displays the normalized, weighted value contributions from five out of seventeen selected value measures. (The values are normalized and weighted in the additive value model, as explained in Section 2.2.2, Eqs. 2.1-2.2.) In the spider plot,

a better value is radially outward from the center, and the shaded area represents higher desirability overall. Architectures C and D have the highest area, as indicated by the legend in Fig. 4.17, so these architectures are likely to have higher additive value. However, only five value measures of interest were selected to be shown in this plot. Future investigations might be able to improve specific architecture designs to maximize value from each measure.

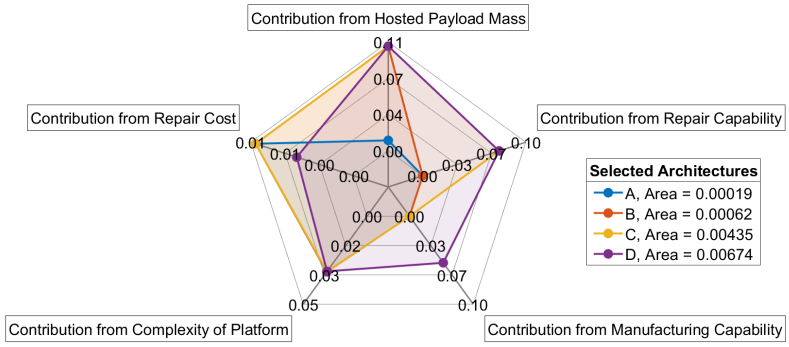


Figure 4.17: Normalized, weighted value contributions from 5/17 total value measures. Better value contributions are radially outward, and higher shaded area indicates higher overall desirability (for these 5 measures).

4.4 Brief Sensitivity Analysis

It is good practice in all modeling activities to run a sensitivity analysis on the many input parameters. While a full sweep of the design variables was performed already in the main analysis (Section 4.3), there were other parameters which could be varied to understand their impact on the results. For this persistent platform study using MAPS, two sets of parameters were subjects of a brief sensitivity analysis: the quantity of value measures and the prioritization of the value measures. This section will present the results.

4.4.1 Case 1: Reduced Quantity of Value Measures

Since there were many value measures present in the main analysis, the first sensitivity case answered the question: How do the architectures compare if the number of value measures are reduced? For Case 1, the quantity of value measures were reduced from seventeen to only eight. Table 4.11 displays the full list of value measures, the original swing weights used in the additive value calculation, and the new swing weights once the quantity was reduced. The rows highlighted in green are the eight value measures that were chosen during the downselect for this sensitivity analysis. The chosen value measures do not just represent those with the highest priority from the main analysis, but instead they were chosen to represent an appropriate spread of the key value measures of interest for the OSAM study.

Figure 4.18 compares the results for value and cost for the original analysis (the black circles) against the sensitivity analysis Case 1 (the blue triangles). For the sensitivity analysis, only twelve architectures were generated to save computational time, and these twelve were all single-module architectures that spanned the breadth of combinations of servicing and manufacturing options. When the quantity of measures was reduced, each architecture had either an increase or reduction in additive value. However, it is important to note that the overall order of which architectures had the most value did not change.

The increase or reduction can only be explained by examining Fig. 4.19, which displays the normalized, weighted contributions of each measure. As described for the original stacked bar chart, Fig. 4.16, the "ideal" column represents the maximum possible score for each measure. Some architectures, such as "S1M1" (Servicing Opt. 1, Manufacturing Option 1), had a reduction in additive value in the sensitivity analysis case because they scored low on measures that now had a higher weight. Also, those architectures did not have other measures to act as "fillers" and make up the value. For example, "S1M1" scored poorly in the measure "presence of repair capability" and "Maintainability", but it did not have high-scoring measures like

Table 4.11: Sensitivity Case 1: Downselected value measures and adjusted weights

	Original Analysis		Sensitivity Analysis – Reduced No. of VMs	
Value Measure	**Matrix Weights**	**Swing Weights**	**Matrix Weights**	**Swing Weights**
Total mass of hosted payloads	100	0.106	100	0.263
Total power for hosted payloads	100	0.106	-	-
Total throughput	90	0.095	-	-
Presence of assembly capability	85	0.090	-	-
Pointing accuracy	80	0.085	-	-
Presence of repair capability	70	0.074	70	0.158
Adaptability	60	0.063	-	-
Maintainability	60	0.063	50	0.132
Scalability	60	0.063	50	0.132
Evolvability	60	0.063	-	-
Presence of manufacturing capability	50	0.053	60	0.158
Position accuracy	40	0.042	-	-
Complexity of platform	30	0.032	30	0.079
Complexity of repairs	20	0.021	-	-
Complexity of upgrade	20	0.021	-	-
Repair cost	10	0.011	10	0.026
Upgrade cost	10	0.011	10	0.026

"total power for hosted payloads" or "presence of assembly capability" to make up for the lost value from the low scores. Conversely, some architectures had an increase in value because they scored high on measures which now had a higher weight. This sensitivity case not only shows that the analysis is sensitive to the quantity of design values but that the inclusion of certain values can change the perception of value of a given architecture. However, the quantity of value measures did not change the overall results of highest-value architectures.

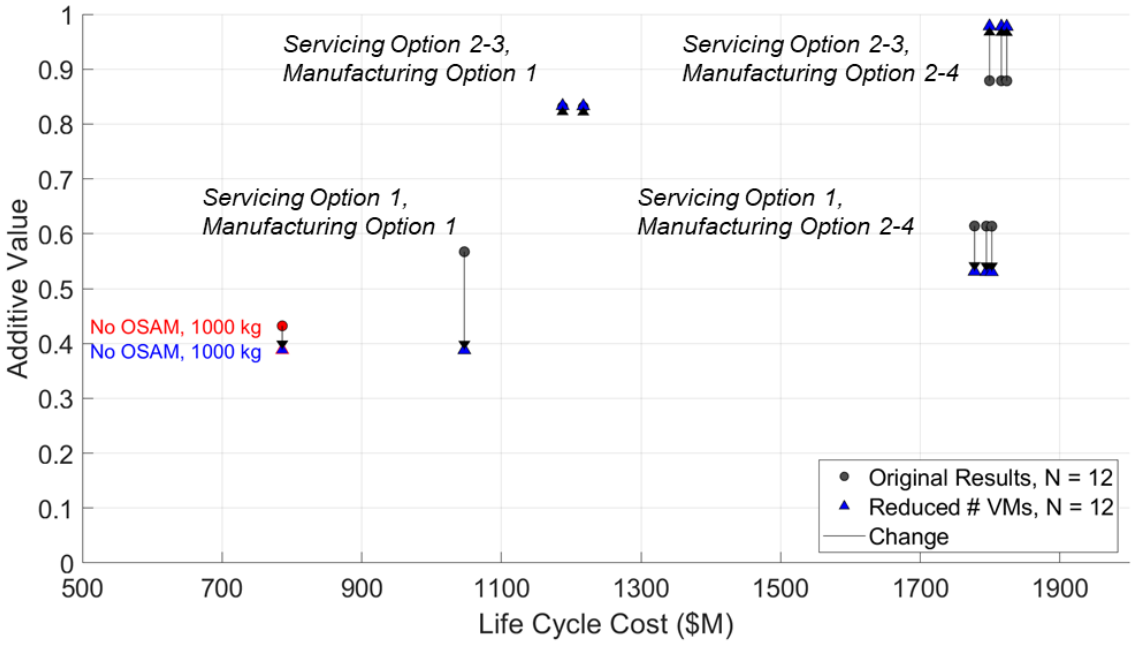


Figure 4.18: When the quantity of value measures are reduced from 17 to 8, architectures may have additive value that is either increased or reduced. A black arrow traces the change for each architecture.

4.4.2 Case 2: Varied Prioritization of Value Measures

Since the swing weights—which represent the priorities of the stakeholders—play an important role in calculating additive value, the second sensitivity case answered the question: How do the architectures compare if the prioritization changes? For Case 2, priority was varied for each value measure in turn, which was accomplished by increasing and also reducing each matrix weight by 20%, then recalculating each swing weight using the method described in Section 4.2.3. Table 4.12 displays the list of value measures and the changed matrix weights. In Case 2, the reduced quantity of value measures from Case 1 was used again to reduce complexity.

Figure 4.20 displays how changing the weight of a single value measure led to both the increase in value for some architectures and the decrease in value for other architectures. This "tornado" chart reveals the sensitivity of the additive value to the matrix weights chosen by the analyst. In tornado charts, the longest bar reflects

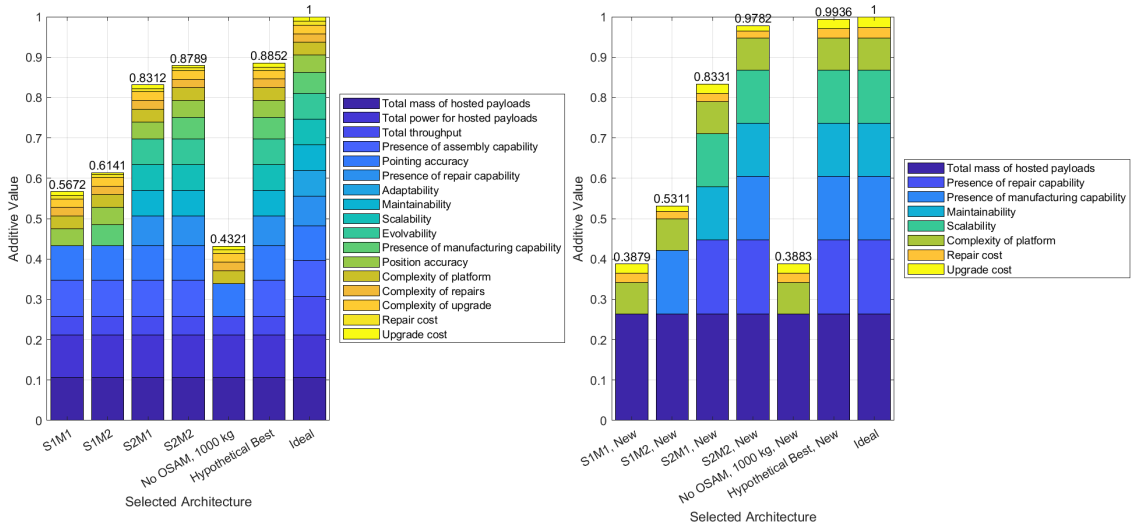


Figure 4.19: Changes to the normalized, weighted contributions from all value measures, when the quantity of measures is reduced from 17 to 8. The stacked contributions provide insight as to why an architecture's additive value was increased or reduced when the quantity of value measures changed.

the measure with the highest impact on the result. However, in this analysis, no single measure has the same impact across all architectures. Another interesting observation is that only two measures—"complexity of platform" and "total mass of hosted payloads"—universally increased the additive value of an architecture if the weight was increased. It is interesting to see how, just as in Fig. 4.19, each unique architecture design combination (i.e., combination of servicing and manufacturing options) impacts the value and the sensitivity of the value in unique ways.

Table 4.12: Sensitivity Case 2: Varied Prioritization (Matrix Weight) of each Value Measure

Value Measure	Matrix Weight, -20%	Matrix Weight, Reference	Matrix Weight, +20%
Total mass of hosted payloads	80	100	120
Presence of repair capability	56	70	84
Maintainability	48	50	72
Scalability	40	50	60
Presence of manufacturing capability	40	60	60
Complexity of platform	24	30	36
Repair cost	8	10	12
Upgrade cost	8	10	12

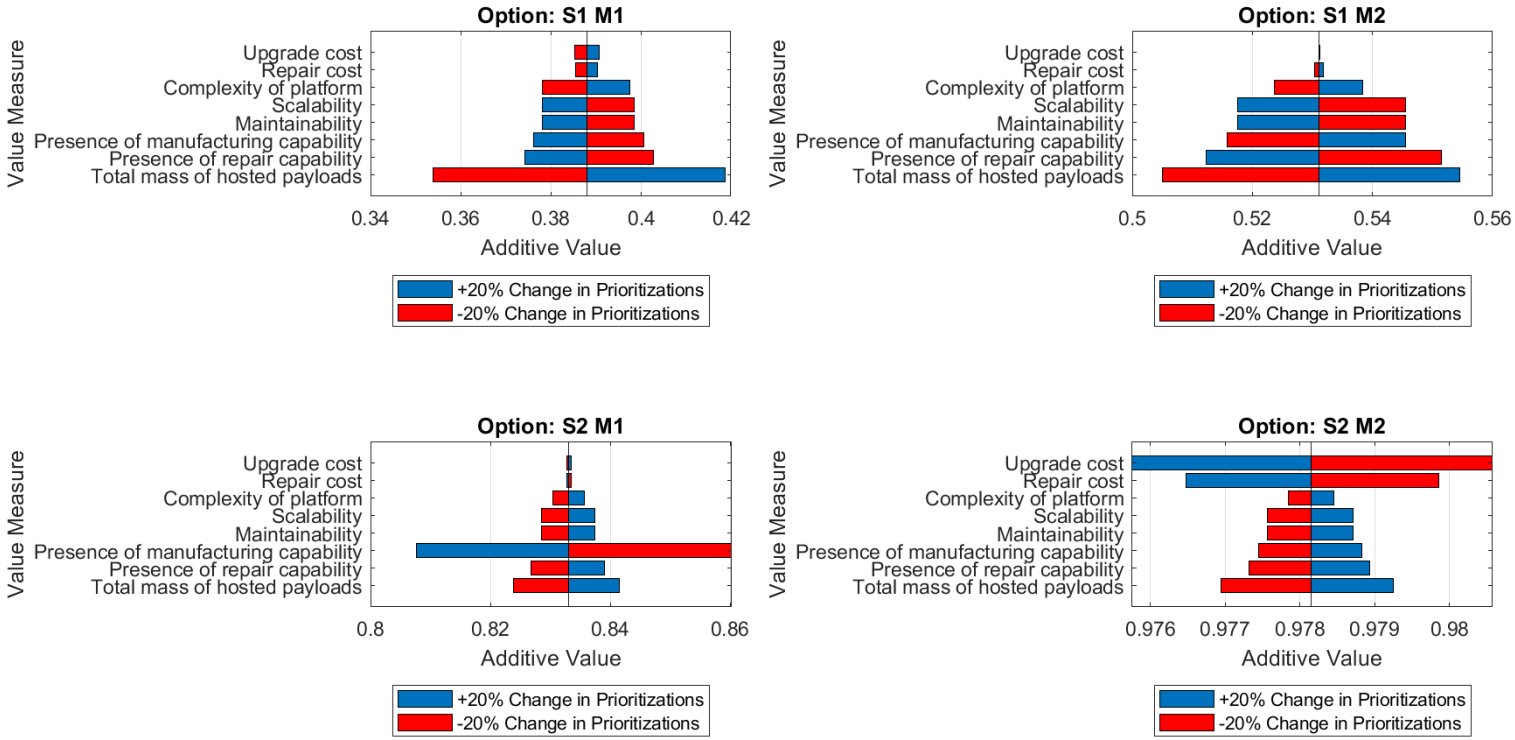


Figure 4.20: Changes to the additive value due to ±20% change in the matrix weight. Each change in weights could led to both the increase in value for some architectures and the decrease in value for other architectures. The size of each bar reflects the sensitivity of additive value to the change in weight.

CHAPTER 5. CONCLUSIONS AND FUTURE WORK

5.1 Summary of Research

This thesis was created to answer two research questions:

- **RQ1:** How can various existing tools for value analysis of space systems or SoS be integrated to improve decision support and to improve usability?
  - **Hypothesis:** By integrating 5 systems analysis tools—encompassing systems design, alternatives enumeration, flexibility, complexity, and life cycle cost—an environment could be created that enables improved evaluation of space systems concepts and improved usability.
- **RQ2:** What design features for an OSAM persistent platform, capable of hosting customer payloads can be varied to best balance value (e.g., flexibility, complexity) and life cycle cost?
  - **Hypothesis:** By varying the level of customer support in the design and varying the On-Orbit Servicing and Manufacturing technologies present on a persistent platform, and by performing analysis with the MAPS environment, we will be able to balance value (e.g., flexibility, complexity) and life cycle cost.

This thesis responded to the research questions by employing both the SoSE discipline and VCA methods to develop a new, integrated environment named Modeling Architectures and Parameterization of Spacecraft (MAPS). It then demonstrated the capabilities of MAPS in a case study focused on a future OSAM market. In Chapter One, a description of the operational context, status quo, and barriers were presented, which begun the Definition phase of SoSE for the demonstration case. In Chapter

Two, a review of the SoSE discipline and VCA was presented along with a review of models for non-traditional architecture value measures. In Chapter Three, the development of the MAPS environment was explained, including explanations of the Space Systems Design, Flexibility, and Complexity tools and an introduction to the MAPS User Guide. In Chapter Four, a case study was presented where the MAPS tools came together with the three-phase DAI process for SoSE. In the case study, a persistent platform was modeled with OSAM capabilities that hosted customer payloads in LEO. Over 18,000 architecture alternatives were evaluated to determine the design characteristics that best balanced value and cost. This study demonstrated the improved usability of the tools in MAPS and the conclusions for decision support.

Although MAPS was first envisioned to provide SoSE and VCA capabilities, which the literature had already proven valuable, the development process also revealed other lessons. First, the importance of having a rigorous value analysis method should be emphasized. Based on the professional experience of the author in internships, having an integrated suite of tools like MAPS would add significant value to organizations that perform systems analysis. A lack of rigorous methods for performing trade-offs, value modeling, and SoS modeling leaves an organization inadequate to achieve maximum value for their stakeholders. Additionally, the approach to trade studies enabled by MAPS—and demonstrated in the case study—facilitates deep dives into non-intuitive results. Running the tools again for follow-ups and sensitivity analyses is not computationally expensive in MAPS. Finally, the focus on usability and adaptability for new users directly impacted the development of MAPS and the User Guide. Therefore, MAPS will be valuable for future studies on emerging space concepts, but the lessons learned can guide other organizations to improve their systems analysis methods.

5.2 Conclusions and Limitations of the Case Study

The results from this case study led to several conclusions about the impact of hosted payload support, OSAM capabilities, and fractionation on a persistent platform architecture. First, there was no noticeable difference between the Base Scenario and the Scalability Scenario. This was likely because there wasn't a significant change in the value measures, so the change was drowned by the large quantity of measures. Next, although architecture alternatives were enumerated by shuffling the payloads/fractionated subsystems between modules, the architectures with only two modules (and in two cases, only one module) always had the lowest cost for approximately the same value.

An increase in hosted payload mass provided an increase in additive value on the order of 10-20%. The mass increase did not require a significant cost increase. The addition of servicing payloads to a platform provided a large increase to the additive value, on the order of 50%, but it did not seem to matter what level of servicing capability. The addition of servicing payloads required a cost increase on the order of 10%. In comparison, the addition of manufacturing payloads provided only around 5% increase to value, but the cost increased approximately 50%. Also, the level of manufacturing onboard the platform did not seem to effect cost or value. In summary, the design characteristics which best balance value and cost on a persistent platform in LEO seem to be high hosted payload mass, some level of in-space servicing, and no in-space manufacturing.

Of course, several limitations of the study exist. First, the Space Systems Design Tool does not use high-fidelity models to address the design of low-level components. During the creation of MAPS, both parametric sizing and publicly available example mass and power data from COTS hardware manufacturers was combined to provide a good foundation for design. However, like all models, they are not perfect representations of the real world and should primarily be used for architecture comparisons, especially during the conceptual phase of the design cycle.

Additionally, the OSAM payloads described in this study are largely notional. The best effort was made to define specific OSAM functions and payloads that made sense for a persistent platform and that allowed a range of options (see Appendix D), but a complete market study was not performed.

Finally, the case study was created using a specific collection of inputs and assumptions. As outlined by the decision hierarchy in Section 4.1.1, several decisions were considered out of the scope of the study, which led to the list of fixed parameters for every architecture (see Appendix B). Each of these fixed parameters could be knobs to turn in other studies, and in those studies the conclusions might change. Any changes to parameters, the status quo drivers, etc. would be unveiled and incorporated during the Definition and Abstraction phases.

Under these limitations, the conclusions presented in the Implementation phase of this case study are truly just demonstrations of what can be learned by using MAPS to apply SoSE and VCA. They cannot provide the ultimate answer to the question, "Is adding OSAM capabilities to a persistent platform valuable despite the cost?" The case study does, however, provide a guide for how MAPS can be used if a decision-maker wishes to answer these questions for their own set of parameters and specific payload information. This study was set up to be a demonstration of the MAPS capabilities, and it does this effectively by describing in detail the steps involved in the Definition, Abstraction, and Implementation phases and by using each of the MAPS tools for space systems design, architecture enumeration, flexibility analysis, complexity analysis, and additive value analysis. Overall, these limitations are lessons learned and can be used for future improvements, as described in Section 5.3.

5.3 Potential Improvements

The limitations of the case study can be lessons learned for future improvements. New studies could explore variations of the persistent platform concept. One idea is to shorten the list of seventeen value measures from the case study presented in this

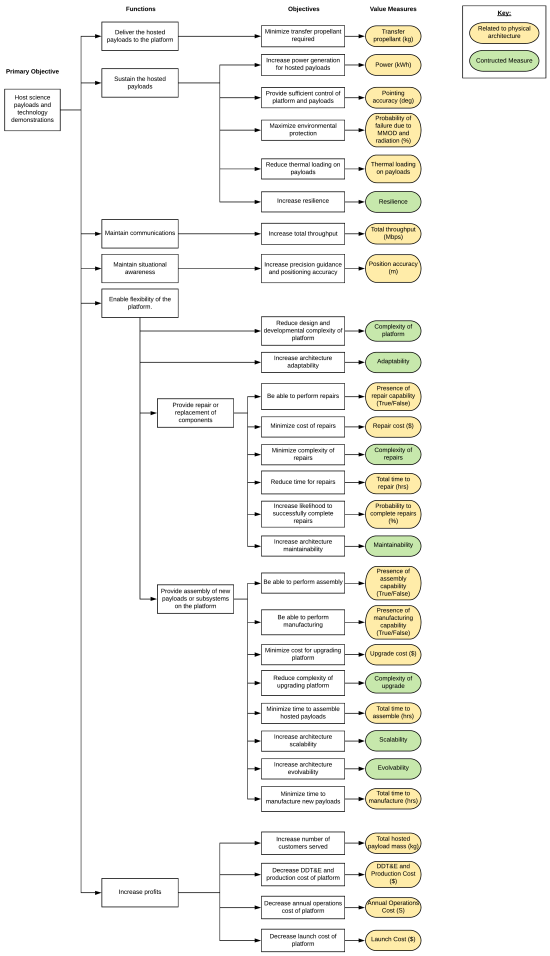


Figure 5.1: Expanded Functional Value Hierarchy for persistent platform architecture with additional value measures for future MAPS capabilities.

thesis. On the other hand, several extra value measures were identified during this study; for example, repair operations time for robotics in comparison to astronaut repair times, such as in (Coffey et al., 2018). Another value measure might focus not on the time an activity takes but on the number of times an activity can be performed, e.g., number of repairs performed. The expanded FVH is provided in Fig. 5.1.

Another improvement would be adding probabilistic analysis. While deterministic analysis shows interesting results in MAPS currently, uncertainty and risk are inherent in the design of new systems, and failing to consider them is a problem (Parnell, 2016).

Therefore, the addition of uncertainty quantification and probabilistic analysis is part of the long-term vision for improving MAPS.

Based on the experience of developing MAPS so far, several potential improvements to MAPS are suggested for the short and long term. In the short term, improved user interfaces could be developed, such as a Graphical User Interfaces (GUI) or data viewer screens. A GUI could be designed to help a user more easily input parameters and navigate the tools. Data viewer screens, such as those in NASA's EXAMINE tool (Komar et al., 2008), could improve immediate data visualization while minimizing the need to interface with the MATLAB code. "Improve alternatives" is a step of the decision management process (Madachy et al., 2019). The GUI and data viewer screens together could enable this step in MAPS.

In the long term, it is envisioned that new tools will be added to MAPS as the need arises. System Operational Dependencies Analysis (SODA) (Guariniello and DeLaurentis, 2017) and System Developmental Dependencies Analysis (SDDA) (Guariniello and DeLaurentis, 2013) both have been suggested as valuable SoSE analysis techniques, and tools exist that could be integrated in MAPS. Other additions could include uncertainty analysis, resiliency or robustness analysis, and qualitative analysis, such as with cascading matrices. Each of these improvements represents a path forward for continuing to develop MAPS into a high-quality modeling and analysis environment that aids current practices with the most up-to-date, integrated tools.

REFERENCES

REFERENCES

Aerospace Corporation (2018). Hive satellites redefine disaggregation. Online; accessed 17 March 2020. `https://aerospace.org/article/hive-satellites-redefine-disaggregation`.

Albright, K. S. (2004). Environmental scanning: radar for success. *Information Management Journal*, 38(3):38–45.

Anderson, M. S., Ewert, M. K., and Keener, J. F. (2018). Life support baseline values and assumptions document.

Bartels, M. (2019). Russia and China are teaming up to explore the Moon. Online; accessed 16 February 2020. `https://www.space.com/russia-china-moon-exploration-partnership.html`.

Brown, O. and Eremenko, P. (2008). Application of value-centric design to space architectures: The case of fractionated spacecraft. In *AIAA SPACE 2008 Conference and Exposition*.

Cavers, M., Fallat, S., and Kirkland, S. (2010). On the normalized laplacian energy and general randić index r-1 of graphs. *Linear Algebra and its Applications*, 433(1):172–190.

Coffey, A., Deardorff, A., Weihing, T., Robertson, B., and Mavris, D. N. (2018). Probabilistic technology selection for in space assembly, manufacture, and repair. In *2018 AIAA SPACE and Astronautics Forum and Exposition*, page 5240.

Collopy, P. D., Bloebaum, C. L., and Mesmer, B. L. (2012). The distinct and interrelated roles of value-driven design, multidisciplinary design optimization, and decision analysis. In *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSM*.

Cross, N. and Roy, R. (2000). Generating alternatives. In *Engineering design methods*, chapter 9, pages 123–138. John Wiley & Sons, New York.

Davendralingam, N., DeLaurentis, D., Fang, Z., Guariniello, C., Han, S. Y., Marais, K. B., Mour, A., and Uday, P. (2014). An analytic workbench perspective to evolution of system of systems architectures. In *CSER*, pages 702–710.

Debus, T. J. and Dougherty, S. P. (2009). Overview and performance of the Front-end Robotics Enabling Near-term Demonstration (FREND) robotic arm. In *AIAA Infotech@Aerospace Conference*.

DeLaurentis, D. A. (2005). Understanding transportation as system-of-systems design problem. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*.

DeLaurentis, D. A., Moolchandani, K. A., and Guariniello, C. (2019). System-of-systems modeling & analysis. Unpublished text for Purdue University AAE 560 Course.

DeLaurentis, D. A., Sindiy, O. V., and Stein, W. B. (2006). Developing sustainable space exploration via a system-of-systems approach. In *AIAA Space and Astronautics Forum*.

di Pippo, S. (2014). The peaceful use of space. Online; accessed 16 February 2020. https://room.eu.com/article/The_peaceful_use_of_space.

DoD, U. (2018). *MIL-STD-881D: Work Breakdown Structures for Defense Materiel Items*.

Eichenberg-Bicknell, E., Wisniewski, M. J., Choi, S. W., , and Westley, D. M. (2009). Using a value-centric tool to optimize lifecycle cost, value and risk of spacecraft architectures. In *AIAA SPACE 2009 Conference and Exposition*.

Erwin, S. (2020). DARPA picks Northrop Grumman as its commercial partner for satellite servicing program. Online; accessed 17 March 2020. https://spacenews.com/darpa-picks-northrop-grumman-as-its-commercial-partner-for-satellite-servicing-p

Espero, T. M. (n.d.). Orbital express: a new chapter in space. Online presentation; accessed 14 May 2020. https://slideplayer.com/slide/10799289/.

European Space Agency (2019a). ESA Space Resources Strategy. Online; accessed 16 February 2020. https://exploration.esa.int/web/moon/-/61369-esa-space-resources-strategy.

European Space Agency (2019b). ESA Strategy for Science at the Moon. Online; accessed 16 February 2020. https://exploration.esa.int/web/moon/-/61371-esa-strategy-for-science-at-the-moon.

Foust, J. (2020). NASA takes Gateway off the critical path for 2024 lunar return. Online; accessed 17 March 2020. https://spacenews.com/nasa-takes-gateway-off-the-critical-path-for-2024-lunar-return/.

Friedenthal, S., Dori, D., and Mordecai, Y. (2019). System modeling concepts. In Cloutier, R., editor, *The Guide to the Systems Engineering Body of Knowledge (SEBoK), v. 2.1*, Hoboken, New Jersey. The Trustees of the Stevens Institute of Technology. Online; accessed 18 March 2020. www.sebokwiki.org. BKCASE is managed and maintained by the Stevens Institute of Technology Systems Engineering Research Center, the International Council on Systems Engineering, and the Institute of Electrical and Electronics Engineers Computer Society.

Grande, M. L., Patel, A. S., Durbin, L. D., and DeLaurentis, D. A. (2020). Modeling architectures and parameterization for spacecraft. In *AIAA Scitech 2020 Forum*.

Grumman, N. (2020). Northrop Grumman successfully completes historic first docking of Mission Extension Vehicle with Intelsat 901 satellite. Online; accessed 17 March 2020. https://news.northropgrumman.com/news/releases/northrop-grumman-successfully-completes-historic-first-docking-of-mission-extens

Guariniello, C. and DeLaurentis, D. A. (2013). Dependency analysis of system-of-systems operational and development networks. *Procedia Computer Science*, 16:265–274.

Guariniello, C. and DeLaurentis, D. A. (2017). Supporting design via the system operational dependency analysis methodology. *Research in Engineering Design*, 28(1):53–69.

Guariniello, C., Fang, Z., O'Neill, W., Ukai, T., Dumbacher, D., and DeLaurentis, D. (2020). Understanding human space exploration as a system-of-systems problem.

Holtta-Otto, K. and de Weck, O. (2007). Degree of modularity in engineering systems and products with technical and business constraints. *CONCURRENT ENGINEERING: Research and Applications*, 15(2):113–921.

Jet Propulsion Laboratory (2018). JPL Strategic Implementation Plan 2018. Online; accessed 17 March 2020. `https://www.jpl.nasa.gov/about/strategic-implementation-plan/capabilities/5/`.

Johnson, E. R., Parnell, G. S., and Tani, S. N. (2013). Perform deterministic analysis and develop insights. In *Handbook of Decision Analysis*, chapter 9, pages 166–226. John Wiley and Sons, New Jersey.

Jones, H. W. (2015). Estimating the life cycle cost of space systems. In *45th International Conference on Environmental Systems*.

Jones, H. W. (2018). Impact of lower launch cost on space life support. In *2018 AIAA SPACE and Astronautics Forum and Exposition*, page 5286.

Keeney, R. L. (1996). *Value Focused Thinking: A Path to Creative Decisionmaking*. Harvard University Press, Cambridge, Massachusetts.

Keeney, R. L., Raiffa, H., et al. (1993). *Decisions with multiple objectives: preferences and value trade-offs*. Cambridge University Press, Cambridge, Massachusetts.

Kemmer, A., Snyder, M., Chen, M., and Dunn, J. (2018). Additive manufacturing of extended structures. US Patent 10,052,820. Online; accessed 14 May 2020. `https://patents.google.com/patent/US10052820B2/en`.

Komar, D., Hoffman, J., Olds, A., and Seal, M. (2008). Framework for the parametric system modeling of space exploration architectures. In *AIAA Space 2008 Conference and Exposition*.

Kugler, J., Cherston, J., Joyce, E. R., Shestople, P., and Snyder, M. P. (2017). Applications for the archinaut in space manufacturing and assembly capability. In *AIAA SPACE and Astronautics Forum and Exposition*, page 5365.

Lafleur, J. and Saleh, J. (2009a). Exploring the f6 fractionated spacecraft trade space with gt-fast. In *AIAA SPACE 2009 Conference and Exposition*.

Lafleur, J. and Saleh, J. (2009b). Gt-fast: a point design tool for rapid fractionated spacecraft sizing and synthesis. In *AIAA SPACE 2009 Conference and Exposition*.

Larson, W. J. and Wertz, J. R. (1992). *Space mission analysis and design*. Microcosm, Inc., Torrance, California.

Maciuca, D. B., Chow, J. K., Siddiqi, A., de Weck, O. L., Alban, S., Dewell, L. D., Howell, A. S., Lieb, J. M., Mottinger, B. P., Pandya, J., et al. (2010). A modular, high-fidelity tool to model the utility of fractionated space systems. *Encyclopedia of Aerospace Engineering*.

Madachy, R., Roedler, G., and Parnell, G. (2019). Decision management. In Cloutier, R., editor, *The Guide to the Systems Engineering Body of Knowledge (SEBoK), v. 2.1*, Hoboken, New Jersey. The Trustees of the Stevens Institute of Technology. Online; accessed 18 March 2020. `https://www.sebokwiki.org/wiki/Decision_Management#:~:text=`. BKCASE is managed and maintained by the Stevens Institute of Technology Systems Engineering Research Center, the International Council on Systems Engineering, and the Institute of Electrical and Electronics Engineers Computer Society.

Made In Space (2017). Made In Space - Archinaut: ULISSES. Online video; accessed 14 May 2020. `https://www.youtube.com/watch?v=wvwXgZhrr-s`.

Made In Space (2019a). Made In Space and Braskem to launch first commercial recycling facility to the Space Station. Online; accessed 17 March 2020. `https://madeinspace.us/press-releases/made-in-space-and-braskem-to-launch-first-commercial-recycling-facility-to-the-s`

Made In Space (2019b). Made In Space awarded NASA contract for robotic manufacturing and assembly flight demo mission. Online; accessed 17 March 2020. `https://madeinspace.us/press-releases/made-in-space-awarded-nasa-contract-for-robotic-manufacturing-and-assembly-fligh`

Made In Space (2019c). SmallSat 2019: Archinaut One Workshop [Recorded Webinar]. Online video; accessed 14 May 2020. `https://www.youtube.com/watch?v=GW3jE-2gSig&feature=emb_rel_end`.

Maghareh, A., Lenjani, A., Dyke, S. J., Marais, K., Whitaker, D., Bobet, A., Ramirez, J., Modiriasari, A., and Theinat, A. (2019). Resilience-oriented design of extraterrestrial habitat systems. In *AIAA Propulsion and Energy 2019 Forum*, page 3972.

Maier, M. W. (1998). Architecting principles for system-of-systems. *Systems Engineering*, 1(4):267–284.

Mann, A. (2019). China's Chang'e program: missions to the Moon. Online; accessed 17 March 2020. `https://www.space.com/43199-chang-e-program.html`.

Mathieu, C. and Weigel, A. L. (2005). Assessing the flexibility provided by fractionated spacecraft. In *AIAA SPACE 2005 Conference and Exposition*.

McCormick, D., Barrett, B., and Burnside-Clapp, M. (2009). Analyzing fractionated satellite architectures using raftimate: a boeing tool for value-centric design. In *AIAA SPACE 2009 Conference and Exposition*.

Moses, J. (2012). The anatomy of large scale systems. *ESD Internal Symposium*.

NASA (2016a). About Restore-L. Online; accessed 17 February 2020. `https://sspd.gsfc.nasa.gov/restore-L.html`.

NASA (2016b). Cubesat Proximity Operations Demonstration (CPOD). Online; accessed 17 March 2020. https://www.nasa.gov/sites/default/files/files/CPOD_FactSheet.pdf.

NASA (2018). NASA Puts In-Space Assembly Robots to the Test. Online; accessed 17 March 2020. https://www.nasa.gov/press-release/langley/nasa-puts-in-space-assembly-robots-to-the-test.

NASA (2019a). NASA announces new partnerships for Commercial Lunar Payload Delivery Services. Online; accessed 16 February 2020. https://www.nasa.gov/press-release/nasa-announces-new-partnerships-for-commercial-lunar-payload-delivery-services.

NASA (2019b). NextSTEP Overview. Online; accessed 16 February 2020. https://www.nasa.gov/content/nextstep-overview.

NASA (2019c). Sending American astronauts to Moon in 2024: NASA accepts challenge. Online; accessed 17 March 2020. https://www.nasa.gov/feature/sending-american-astronauts-to-moon-in-2024-nasa-accepts-challenge.

NASA Executive Cost Analysis Steering Group and others (2000). Appendix B: Work Breakdown Structure (WBS). In *NASA Cost Estimating Handbook, Version 4.0*. NASA.

Neema, K., Tamaskar, S., Guariniello, C., and DeLaurentis, D. (2018). Complexity and flexibility enabled model based design framework for space system design. In *2018 AIAA SPACE and Astronautics Forum and Exposition*.

Nilchiani, R. and Hastings, D. E. (2005). *Measuring the value of space systems flexibility: a comprehensive six-element framework*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Ogilvie, A., Allport, J., Hannah, M., and Lymer, J. (2007). Autonomous satellite servicing using the Orbital Express demonstration manipulator system. In *DARPA*.

Parnell, G. S., editor (2016). *Trade-Off Analytics: Creating and Exploring the System Tradespace*. John Wiley and Sons, Hoboken New Jersey.

Patane, S., Schomer, J., and Snyder, M. (2018). Design reference missions for archinaut: A roadmap for in-space robotic manufacturing and assembly. In *2018 AIAA SPACE and Astronautics Forum and Exposition*, page 5188.

Piskorz, D. and Jones, K. L. (2018). On-orbit assembly of space assets: A path to affordable and adaptable space infrastructure. Online; accessed 18 March 2020. https://aerospace.org/sites/default/files/2018-05/OnOrbitAssembly_0.pdf.

Pugliese, A. and Nilchiani, R. (2017). Complexity analysis of fractionated spacecraft architectures. In *AIAA SPACE and Astronautics Forum and Exposition*.

Raz, A. K. and DeLaurentis, D. A. (2017). System-of-systems architecture metrics for information fusion: A network theoretic formulation. In *AIAA Information Systems-AIAA Infotech @ Aerospace*.

Raz, A. K., Kenley, C. R., and DeLaurentis, D. A. (2018). System architecting and design space characterization. *Systems Engineering*, 21(3):227–242.

Richardson, G., Penn, J., and Collopy, P. D. (2010). Value-centric analysis and value-centric design. In *AIAA SPACE 2010 Conference and Exposition*.

Roesler, G. (2016). Robotic Servicing of Geosynchhronous Satellites (RSGS) Proposers Day. Presentation online; accessed 17 March 2020. https://www.darpa.mil/attachments/RSGSProposersDaySlideDeck.PDF.

Ross, A. (2003). *MULTI-ATTRIBUTE TRADESPACE EXPLORATION WITH CONCURRENT DESIGN AS A VALUE-CENTRIC FRAMEWORK FOR SPACE SYSTEM ARCHITECTURE AND DESIGN*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Ross, A. M., Hastings, D. E., Warmkessel, J. M., and Diller, N. P. (2004). Multi-attribute tradespace exploration as front end for effective space system design. *JOURNAL OF SPACECRAFT AND ROCKETS*, 41(1):20–28.

Schwarz, R. E., Helmer, R. E., Briggs, P. A., Lymer, J. D., Tadros, A. H., and Turner, A. E. (2018). Self-assembling persistent space platform. US Patent US2018/0093786 A1. https://patents.google.com/patent/US20180093786A1/en.

Sequin, C. (2019). Cislunar blueprint to propel space outreach for the next 50 years. Online; accessed 13 May 2020. https://www.purdue.edu/newsroom/releases/2019/Q3/purdue-engineerings-cislunar-initiative-creates-five-step-blueprint-to-propel-spa html.

Sierra Nevada Corporation (2020). Space technologies product catalog. Online; accessed 14 May 2020. https://www.sncorp.com/what-we-do/space-technologies-subsystems/.

Sinha, K. and de Weck, O. L. (2013). Structural complexity quantification for engineered complex systems and implications on system architecture and design. In *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection.

SpaceX (2020). Capabilities & services. Online; accessed 19 March 2020. https://www.spacex.com/about/capabilities.

Spencer, D. A. (2015). *AUTOMATED TRAJECTORY CONTROL FOR PROXIMITY OPERATIONS USING RELATIVE ORBITAL ELEMENTS*. PhD thesis, Georgia Institute of Technology, Atlanta, Georgia.

Sullivan, B. R., Parrish, J. C., and Roesler, G. (2018). Upgrading in-service spacecraft with on-orbit attachable capabilities. In *AIAA Space and Astronautics Forum and Exposition*.

Tamaskar, S. (2014). *MANAGING COMPLEXITY OF AEROSPACE SYSTEMS*. PhD thesis, Purdue University, West Lafayette, Indiana.

Tamaskar, S., Neema, K., and DeLaurentis, D. (2014). Framework for measuring complexity of aerospace systems. *Research in Engineering Design*, 25(2):125–137.

Tethers Unlimited (2017). Constructable Platform Enabled by PODS. Online; accessed 17 March 2020. https://www.sprsa.org/sites/default/files/presentations/Hoyt_Transformation%20Technologies%20for%20Space%20and%20Defense%20Missions.pdf.

Ullman, D. G. (2003). Concept generation. In *The mechanical design process*, chapter 7, pages 137–141. McGraw-Hill, Boston.

United Nations Office for Outer Space Affairs (n.d.). Treaty on principles governing the activities of states in the exploration and use of outer space, including the moon and other celestial bodies. Online; accessed 16 February 2020. https://www.unoosa.org/oosa/en/ourwork/spacelaw/treaties/introouterspacetreaty.html.

US Air Force, Space and Missile Systems Center (n.d.). Chapter 3: Uscm database development. Online; accessed 06 May 2020. https://www.uscmonline.com/.

Wall, M. (2019). 50 years after apollo 11, a new moon rush is coming. Online; accessed 16 February 2020. https://www.space.com/moon-exploration-plans-nasa-india-china-and-more.html.

Wang, J. R. (2017). New space policy directive calls for human expansion across solar system. Online; accessed 17 March 2020. https://www.nasa.gov/press-release/new-space-policy-directive-calls-for-human-expansion-across-solar-system.

Wertz, J. R., Everett, D. F., and Puschell, J. J. (2011). *Space mission engineering: the new SMAD*. Microcosm Press.

Wikipedia (2019). Registration convention. Online; accessed 16 February 2020. https://en.wikipedia.org/wiki/Registration_Convention.

Wikipedia (2020). Space liability convention. Online; accessed 16 February 2020. https://en.wikipedia.org/wiki/Space_Liability_Convention.

Williams, A. (2015). Cubesat proximity operations demonstration (cpod) vehicle avionics and design. Online; accessed 14 May 2020. https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=3291&context=smallsat.

Williams, P., Dempsey, J., Hamill, D., Rodgers, E., Mullins, C., Gresham, E., and Downs, S. (2018). Space Science Technology Partnership Forum: Value Proposition, Strategic Framework, and Capability Needs for In-Space Assembly. In *2018 AIAA SPACE and Astronautics Forum and Exposition*.

APPENDICES

APPENDIX A. COMPONENT SELECTIVITY AND CONNECTIVITY TABLES

Table A.1: Component Selectivity Rules.

Subsystem	**Component Library**	**Selectivity Rules**
ADCS	• Software_adcs • Horizon (Earth) sensor • Sun sensor • Magnetometer • Star sensor • Momentum wheel • Reaction wheel • Magnetic torquer • Inertial Measurement Unit (IMU)	• Software_adcs should always be part of the subsystem. • If module mass $>$ 10 kg, then subsystem contains a GPS. • If control accuracy requirements (CAR) of the module $\geq 5°$, then subsystem contains 4 sun sensors, 1 magnetometer and 1 momentum wheel. • If $1° < \text{CAR} < 5°$, then subsystem contains 4 sun sensors, 2 horizon (Earth) sensors, 1 magnetometer, 3 reaction wheels, and 3 magnetic torquers.

Continued on next page

Table A.1 *continued*

Subsystem	**Component Library**	**Selectivity Rules**
ADCS *contd.*	• GPS	• If $0.1^\circ < \text{CAR} < 1^\circ$, then subsystem contains 4 star trackers, 2 horizon sensors, 1 magnetometer, 1 (optional) IMU, 3 reaction wheels, and 3 magnetic torquers. • If $\text{CAR} < 0.1^\circ$, then subsystem contains 4 star trackers, 1 magnetometer, 1 (better) IMU, 3 reaction wheels, and 3 (better) magnetic torquers.
Avionics	• Software_avionics • Command unit	• Subsystem should include one of each component.

Continued on next page

Table A.1 *continued*

Subsystem	**Component Library**	**Selectivity Rules**
Comm	• Software_comm • Antenna • Transceiver	• Subsystem should include one of each component.

Continued on next page

Table A.1 *continued*

Subsystem	Component Library	Selectivity Rules
ECLSS	• Software_eclss • Atmospheric Control and Supply components • Temperature and Humidity Control components • Atmosphere Revitalization components • Fire Detection components • Vacuum Services components • Water Recovery and Management components • EVA components	• Subsystem should include all components. • Quantities of components and spares (as well as associated masses) are detailed in NASA's Life Support Baseline Values and Assumptions Document based on state-of-the-art ISS system (Anderson et al., 2018). • Two oxygen and two nitrogen tanks are included among the Atmosphere Control and Supply components (Anderson et al., 2018), but size of tanks must be determined by mission requirements.

Continued on next page

Table A.1 *continued*

Subsystem	**Component Library**	**Selectivity Rules**
Payload	*Varies—defined by user*	All payloads are defined by user and may or may not include separate components. At minimum, payloads are defined by: dry mass, power requirement, and pointing accuracy. Mechanisms should be defined as payloads by user if applicable.
Power	• Software_power • Batteries • Battery charger • Solar array	• Subsystem should include all components. • If module CAR ¡ 5°, then subsystem contains panel-mounted solar arrays. Otherwise, the arrays are body-mounted. • Wiring mass is sized based on a percentage (2.5%) of module dry mass (Wertz et al., 2011).

Continued on next page

Table A.1 *continued*

Subsystem	Component Library	Selectivity Rules
Power *contd.*	• Power control unit (PCU) • Array calibration motors • Wiring	• Subsystem contains one PCU and one motor per protruding solar array arm (i.e. two motors if solar arrays are panel mounted). • Total solar array area and mass is calculated to support total module power requirements plus losses from wiring and conversion, and margin. Time in daylight vs. in eclipse and array degradation is considered. • Batteries are standard Lithium Ion type batteries. They are sized to support full module power requirements during eclipse, with a lower bound on the depth of discharge (30%) (Wertz et al., 2011).

Continued on next page

Table A.1 *continued*

Subsystem	Component Library	Selectivity Rules
Propulsion	• Software_prop • Main and Reaction Control System (RCS) thrusters • Pressure regulators • Pressurized gas tank • Propellant tank • Hydrazine propellant	• Subsystem should include at least one of all components. • By default, the subsystem contains one main thruster and 12 RCS thrusters. Each are sized based on public data for "MONARC" hydrazine thrusters from MOOG. • Tank mass and propellant mass are sized based on the required delta-v, including a standard level for station-keeping and margin (10%) (Wertz et al., 2011).

Continued on next page

Table A.1 *continued*

Subsystem	**Component Library**	**Selectivity Rules**
Thermal	• Software_therm • Multi-Layer Insulation (MLI) • Heaters • Radiator • Temperature sensors • Heat sink	• By default, subsystem includes passive and active thermal management components. • MLI mass is estimated based on total module dry mass. • Subsystem contains one heater, but the heater mass and power requirement is sized based on total module mass. Subsystem also contains one heater dedicated to the payloads. • Subsystem contains one temperature sensor per heater, one heat sink, and one radiator.

Table A.2: Component Connectivity Rules.

Subsystem	**Intra-Subsystem Connectivity Rules**	**Inter-Subsystem Connectivity Rules**
ADCS	• All the sensors have a data link to software_adcs to send their sensed data. • Software_adcs has a data link to all the actuators to send information on the forces and torques required for maintaining the control accuracy.	• Software_adcs has a data link to the software_avionics component, and it also receives information from software_avionics. • Power is received from power subsystem (PCU).
Avionics	• The command unit has a data link to send and receive information from software_avionics.	• Software_avionics has a data link to send and receive information from the software components of all other subsystems. • Power is received from power subsystem (PCU).

Continued on next page

Table A.2 *continued*

Subsystem	**Intra-Subsystem Connectivity Rules**	**Inter-Subsystem Connectivity Rules**
Comm	• The antenna passes incoming external information through the transceiver to software_comm. • Software_comm passes incoming internal information through the transceiver to the antenna.	• Software_comm has a data link to send and receive information from software_avionics. • Power is received from power subsystem (PCU).

Continued on next page

Table A.2 *continued*

Subsystem	Intra-Subsystem Connectivity Rules	Inter-Subsystem Connectivity Rules
ECLSS	• Software_eclss provides information transfer between the component collections (e.g. between Atmosphere Control & Supply and Vacuum Services). • Temperature & Humidity Control (THC) interfaces with Water Recovery Management (WRM). • Fire Detection communicates with software_eclss to interface with Atmospheric Control & Supply (ACS) and Vacuum Services (VS) in the case of an emergency. • ACS interfaces with Atmosphere Revitalization (AR) and VS.	• Software_eclss has a data link to send and receive information from software_avionics. • Power is received from power subsystem (PCU).

Continued on next page

Table A.2 *continued*

Subsystem	**Intra-Subsystem Connectivity Rules**	**Inter-Subsystem Connectivity Rules**
Payload	*Varies—defined by user*	• *Varies—defined by user* • Software_pay should have a data link to send and receive information from software_avionics. • Power is received from power subsystem (PCU).

Continued on next page

Table A.2 *continued*

Subsystem	Intra-Subsystem Connectivity Rules	Inter-Subsystem Connectivity Rules
Power	• Software_power has a data link to each motor to send information on the desired orientation of the arrays. Software_power also distributes power to the motors. • Each motor is structurally connected to solar arrays. • Solar arrays absorb solar power and deliver that power to the battery charger. • The battery charger is linked to each battery, and the batteries are linked to the PCU. • The PCU distributes power to power subsystem components via software_power. • Software_power has a data link to the PCU to provide information on the power distribution to different subsystems.	• Software_power has a data link to send and receive information from software_avionics. • Electrical power is distributed to all relevant subsystems (via software components).

Continued on next page

Table A.2 *continued*

Subsystem	**Intra-Subsystem Connectivity Rules**	**Inter-Subsystem Connectivity Rules**
Propulsion	• Software_prop controls the thrusters using a data link. • Software_prop controls the electric pressure regulators using a data link, and the regulator is linked between the pressurized gas tank and propellant tank. • The pressurized gas tank passes fluid to the propellant tank via the electric pressure regulator, and the propellant tank passes fluid to each thruster when needed.	• Software_prop has a data link to send and receive information from software_avionics. • Power is received from power subsystem (PCU).

Continued on next page

Table A.2 *continued*

Subsystem	**Intra-Subsystem Connectivity Rules**	**Inter-Subsystem Connectivity Rules**
Thermal	• Each heater has a thermal link to a temperature sensor, and the temperature sensors have a data link to software_therm. • Each heater has a data link to software_therm to send and receive information. • The heat sink has a thermal link to the radiator.	• Software_therm has a data link to send and receive information from software_avionics. • Power is received from power subsystem (PCU).

APPENDIX B. FIXED PARAMETERS FOR ALL ARCHITECTURES

The following fixed parameters were used when generating all architectures for the case study to reduce the number of design variables.

- The platform will be designed with all the traditional spacecraft subsystems to support the customer payloads
  - The platform will be designed to generate 1,000 W of electrical power to support the hosted customer payloads, plus sufficient power generation for the support subsystems. The platform will also be designed with sufficient power storage to last through eclipse periods.
  - The platform will be designed for a control pointing accuracy of 1 °to support the hosted customer payloads. However, if any subsystem or payload requires better accuracy, this will be provided.
  - The platform will utilize Ka-band antennas and transceivers to provide communications downlink for customer payloads at a rate of 500 Mbps. Therefore, all architectures will provide the same total throughput (which is a value measure).
  - The platform will utilize S-band antennas and transceivers to provide communications uplink for platform and payload command and control.
- The launch of all platform modules, the service module, and subsequent launches will be purchased at a "pay per kg" price.
  - This is done for consistency as well as to acknowledge that many individual modules may be able to take advantage of cheaper ride-along services.
- A Service Module will loiter in LEO to deliver all modules from the altitude achieved by the launch vehicle to the platform location.

  - The Service Module is the active agent to perform proximity and rendezvous operations and berthing with each module, which are passive.
  - The Service Module is the active agent to perform proximity and rendezvous operations with the platform, which is passive.
  - The Service Module and a robotic arm on the platform will together enable docking of modules with the platform.
- Any servicing will involve replacement or manufacturing of full subsystems.
  - Evaluation and identification of subsystems will be performed using a combination of an onboard catalog, imaging systems, and image recognition systems.
  - Subsystems or payloads may be delivered in new modules that will be docked to the platform and will await unpacking/individual installations.
  - Servicing will be done by the platform (not the service module).

APPENDIX C. ASSESSMENT OF MATRIX WEIGHTS FOR VALUE MEASURES

A swing weight matrix is composed of each value measure, matrix weights, and swing weights. This is one method to developing swing weights, in which each value measure is assigned a location in the swing weight matrix based on two factors: importance of the measure to the decision and impact of the range of the measure. Next, each measure is given a numerical value, called a matrix weight, in a specific order (Johnson et al., 2013). The highest matrix weight is given to the value measure in the top left corner, and each weight is assigned in order until the lowest weight is given to the value measure in the bottom right corner. This order is displayed in Fig. C.1.

Impact of the Range of the VM	Mission Critical		Enabling		Enhancing	
	Value Measure	Matrix Weight	Value Measure	Matrix Weight	Value Measure	Matrix Weight
Significant Impact	Total mass of hosted payloads	100	Presence of assembly capability	85	Adaptability	60
	Total power for hosted payloads	100			Maintainability	60
					Scalability	60
					Evolvability	60
Medium Impact	Total throughput	90	Presence of repair capability	70	Position accuracy	40
Minimal Impact	Pointing accuracy	80	Presence of manufacturing capabilities	50	Complexity of platform	30
					Complexity of repairs	20
					Complexity of upgrade	20
					Repair cost	10
					Upgrade cost	10

A B1 C1 B2 C2 D1 C3 D2 E1 E2 E3

Figure C.1: Example progression of assigning matrix weights to each value measure.

APPENDIX D. FULL PROCESS FOR DEFINING OSAM PAYLOADS

Chapter Four reviews the process by which new OSAM payloads were defined for the case study. This process relies on the morphological matrix created to generate design options, which was provided in Tables 4.2-4.3. A complete design, based on a morphological matrix, is defined by combining one means (in a column) for each function (in a row) (Cross and Roy, 2000). Understanding this, the process for the OSAM payloads therefore included:

1. The functions in the morphological chart were sorted to identify those that were related to each category of assembly, servicing, and manufacturing.
2. A set of means combinations was put together for each category.
3. Each combination was translated into the specific technologies/payloads needed for the means.

The following sections will provide the results of each of the three steps in the three OSAM categories. Finally, additional details and sources will be provided about each of the specific OSAM technologies used in the case study.

D.1 Assembly Options

For Assembly, the sorting in Step 1 and the combinations created in Step 2 are displayed in Table D.1. The specific technologies defined to serve these means are displayed in Table D.2. In the final step, all of these technologies were combined into a single payload, displayed in Table D.3. This final combination was done to reduce the modeling complexity, so MAPS would only need to track a single payload ID for assembly rather than three individual payloads.

Table D.1: Based on the morphological matrix, functions that fall within the assembly category and combinations of means to form one assembly option.

Category	Function	Option 1 - Passive platform modules during docking with assist from Service Module
Assembly	Assemble platform modules	Every module is passive, and service module is active agent to lead docking of each module. SM has RPO payload, e.g. imaging system.
	Perform proximity operations with platform	Platform is passive during approach
	Perform rendezvous operations with platform	Platform is passive for docking, and modules/SM are active
	Dock or berth to platform	Robotic arm on SM or platform guides module with grappling points and coupling mechanism

Table D.2: Individual technologies defined to serve the means for Assembly.

Payload	Properties			
	Mass (kg)	Power Req. (W)	Pointing Accuracy Req. (deg)	Position Accuracy (m)
Rendezvous and Proximity Operations Package (RPO)[1]	6.5	27.5	0.15	1.0 (provided)
Structural, Power, and Data Port – Active (SPA)[2]	6.15	15	0.15	0.1 (required)
Structural, Power, and Data Port – Passive (SPP)[2]	0.575	0	5	0.1 (required)

Table D.3: Payloads for Assembly that are input to MAPS.

Payload	Payload ID	Properties					
		Dry Mass (kg)	Wet Mass (kg)	Power Req. (W)	Pointing Accuracy Req. (deg)	Position Accuracy (m)	NICM Instrument Category
Assembly Package (AP) - 2x SPA, 2x SPP	1	13.45	13.45	30	0.15	0.1	1

D.2 Servicing Options

For Servicing, the sorting in Step 1 and the combinations created in Step 2 are displayed in Table D.4. The specific technologies defined to serve these means are displayed in Table D.5. In the final step, these technologies were combined into a smaller quantity of payloads, displayed in Table D.6.

Table D.4: Based on the morphological matrix, functions that fall within the servicing category and combinations of means to form one servicing option.

Category	Function	Option 1 – None	Option 2 – Payload or Module Install Only (Platform)	Option 3 – Payload Install and Subsystem Swap (Platform)
Servicing	Identify location to install payloads	n/a – payloads integrated on the ground	Onboard catalog, image recognition software, and imaging systems on platform	Onboard imaging systems, image recognition software, and catalog on platform
	Install payloads	n/a – payloads integrated on the ground	Robotic arm on platform (RAI)	Robotic arm on platform (RAS)
	Identify sufficient resources	n/a – no maintenance capability	n/a – no maintenance capability	Onboard imaging systems, image recognition software, and catalog on platform
	Identify location to store resources	n/a – no maintenance capability	n/a – no maintenance capability	Onboard imaging systems, image recognition software, and catalog on platform
	Store resources	n/a – no maintenance capability	n/a – no maintenance capability	Module remains docked to platform and no unpacking necessary yet
	Identify location of failed subsystem	n/a – no maintenance capability	n/a – no maintenance capability	Onboard imaging systems, image recognition software, and catalog on platform
	Remove failed subsystem and dispose of it	n/a – no maintenance capability	n/a – no maintenance capability	Robotic arm on platform (RAS)
	Install new subsystem	n/a – no maintenance capability	n/a – no maintenance capability	Robotic arm on platform (RAS)

Table D.5: Individual technologies defined to serve the means for Servicing.

Payload	Payload ID	Properties					
		Dry Mass (kg)	Wet Mass (kg)	Power Req. (W)	Pointing Accuracy Req. (deg)	Position Accuracy (m)	NICM Instrument Category
Robotic Arm for Installation (RAI + SPA)	2	83.15	83.15	145	0.15	0.1	1
Robotic Arm for Subsystem Servicing (RAS + SPA)	3	108.15	108.15	188	0.15	0.01	1
Onboard imaging systems, image recognition software, and catalog (OIS)	4	6.5	6.5	27.5	0.1	nan	1

Table D.6: Payloads for Servicing that are input to MAPS.

Payload	Properties			
	Mass (kg)	Power Req. (W)	Pointing Accuracy Req. (deg)	Position Accuracy (m)
Structural, Power, and Data Port – Active (SPA)	6.15	15	0.15	0.1 (required)
Onboard imaging systems, image recognition software, and catalog (OIS) [1]	6.5	27.5	0.1	n/a
Robotic Arm for Installation (RAI) [2]	77	130	n/a	0.1 (provided)
Robotic Arm for Subsystem Servicing (RAS) [3]	102	173	n/a	0.01 (provided)

D.3 Manufacturing Options

For Manufacturing, the sorting in Step 1 and the combinations created in Step 2 are displayed in Table D.7. The specific technologies defined to serve these means are displayed in Table D.8. In the final step, all of these technologies were combined into only four payloads, displayed in Table D.9.

Table D.7: Based on the morphological matrix, functions that fall within the servicing category and combinations of means to form four manufacturing options.

Category	Function	Option 1 – None	Option 2 – Manufacture Platform Structures	Option 3 – Manufacture Structures + Solar Arrays	Option 4 – Manufacture Structures + Solar Arrays + Electronics
Manufacturing	**Identify location to install payloads**	n/a – no manufacturing capability	Onboard catalog, image recognition software, and imaging systems on platform	Onboard catalog, image recognition software, and imaging systems on platform	Onboard catalog, image recognition software, and imaging systems on platform
	Access resources for manufacturing	n/a – no manufacturing capability	Robotic arm on platform (RAS)	Robotic arm on platform (RAS)	Robotic arm on platform (RAS)
	Manufacture support subsystem	n/a – no manufacturing capability	Additive Manufacturing (AM) of extended structures	Additive Manufacturing (AM) of structures + assembly of flexible blanket solar arrays, with structures	Additive Manufacturing (AM) + assembly of flexible blanket solar arrays, with structures + AM of electronics
	Manufacture payload	n/a – no manufacturing capability	n/a – payloads manufactured on the ground	n/a – payloads manufactured on the ground	n/a – payloads manufactured on the ground
	Join components (screw, latch, glue, etc.)	n/a – no manufacturing capability	Subsystem servicing end effector on robotic arm	Subsystem servicing end effector on robotic arm	Subsystem servicing end effector on robotic arm
	Verify and validate successful manufacture of subsystem	n/a – no manufacturing capability	Laser end effector for inspection and verification	Laser end effector for inspection and verification	Laser end effector for inspection and verification

Table D.8: Individual technologies defined to serve the means for Servicing.

Payload	Properties			
	Mass (kg)	Power Req. (W)	Pointing Req. (deg)	Position Accuracy (m)
Structural, Power, and Data Port – Active (SPA)	6.15	15	0.15	0.1 (required)
Robotic Arm for Subsystem Servicing (RAS)	102	173	n/a	0.01 (provided)
Laser End Effector for V&V (LVV)	5	5	n/a	0.01 (required)
Additive Manufacturing Machine (AMM)[1]	50	300	n/a	n/a
Thermoplastic Feedstock for Structures (TFS)[2]	11.5	0	n/a	n/a
Flexible Blanket Solar Arrays (FBS)[3]	20	0	n/a	n/a
Feedstock for Electronics (FEL)	10	0	n/a	n/a

Table D.9: Payloads for Servicing that are input to MAPS.

Payload	Payload ID	Properties					
		Dry Mass (kg)	Wet Mass (kg)	Power Req. (W)	Pointing Accuracy Req. (deg)	Position Accuracy (m)	NICM Instrument Category
Robotic Arm for Subsystem Servicing (RAS + SPA)	3	108.15	108.15	188	0.15	0.01	1
Manufacturing Package 2 (SPA + LVV + AMM + TFS)	5	72.65	72.65	320	nan	0.01	1
Manufacturing Package 3 (SPA + LVV + AMM + TFS + FBS)	6	92.65	92.65	320	nan	0.01	1
Manufacturing Package 4 (SPA + LVV + AMM + TFS + FBS)	7	102.65	102.65	320	nan	0.01	1

D.4 Details and Sources for the Individual OSAM Technologies

The three basic properties for payloads—including mass, power requirement, and pointing accuracy requirement—have already been defined in the tables for Assembly, Servicing, and Manufacturing. However, two additional sets of information were also input to MAPS for each payload: the instrument category for the NASA Instrument Cost Model (NICM) and the adjacency matrix. In this section, the adjacency matrices, instrument cost categories, and sources will be provided for the individual OSAM technologies.

The limitations of this work must be noted. Significant effort was made to define each OSAM payload. However, specific data was in most cases either proprietary or unavailable to the public online, so the payloads remain largely notional.

D.4.1 Assembly Technologies

Rendezvous and Proximity Operations Package (RPO)

The first technology to enable on-orbit assembly was the Rendezvous and Proximity Operations Package (RPO). RPO was included on only the Service Module in each architecture. Its purpose was to enable the Service Module to perform rendezvous and proximity operations with each module and also with the platform. There have been several missions in recent years to demonstrate proximity operations and in some cases docking (Spencer, 2015). However, it is very difficult to find specific information online about the sensors and docking mechanisms onboard those spacecraft.

RPO was modeled after the RPO package of the same name on the CubeSat Proximity Operations Demonstration (CPOD) mission, developed in 2015 by Tyvak (Williams, 2015). CPOD contained several relevant, lightweight components that are included in the RPO package: Wide Field Visible Imager, Infrared Imagers (x2), Relative Position Estimation Processor, Maneuver Planning Processor (part of CPOD's ADCS subsystem), and Optical Target Aid. A Tyvak presentation online provided mass, power, and quantity data for each of these components as well as a connectivity diagram (Williams, 2015). The connectivity diagram was used to create the RPO adjacency matrix, which is displayed in Fig. D.1.

	WF Imager	IR Imager	IR Imager	Optical Target Aid	RPE Processor (*Software_RPO*)
WF Imager	0	0	0	0	1
IR Imager	0	0	0	0	1
IR Imager	0	0	0	0	1
Optical Target Aid	0	0	0	0	0
RPE Processor (*Software_RPO*)	1	1	1	1	0

Figure D.1: Adjacency matrix for RPO payload.

There should also be links from the components within RPO to components in other subsystems, as shown in Fig. D.2. When an adjacency matrix is made for a whole module by combining subsystem matrices, these links must be added—e.g., by writing a new script in MAPS. In most payloads developed in this study, external links include a data link to and from the module's "Software_Avionics" component and an electrical power link from the module's Power Control Unit. If the module does not have a power subsystem, then it must receive electrical power through a Structural, Power, and Data Port – Passive (SPP), as indicated in Fig. D.2.

	Payload Adjacency Matrix					External Links		
	WF Imager	IR Imager	IR Imager	Optical Target Aid	RPE Processor (*Software_RPO*)	Software_ Avionics	Power Control Unit	Software _SPP
WF Imager	0	0	0	0	1	0	0	0
IR Imager	0	0	0	0	1	0	0	0
IR Imager	0	0	0	0	1	0	0	0
Optical Target Aid	0	0	0	0	0	0	0	0
RPE Processor (*Software_RPO*)	1	1	1	1	0	1	0	0
Software_Avionics	0	0	0	0	1			
Power Control Unit	0	0	0	0	1			
Software_SPP	0	0	0	0	1			

External links in same module

External links if module has no power subsystem

External links if module has power subsystem

Figure D.2: Adjacency matrix for RPO payload with external links to other subsystems.

Structural, Power, and Data Port – Active (SPA) and Passive (SPP)

The next two payloads to enable on-orbit assembly are the Structural, Power, and Data Ports. These are the structural docking ports, and they also transfer power and data between modules. Every module in the case study was required to have two active (SPA) and two passive (SPP) units as part of the "Assembly Package" payload (see Table D.3).

The SPA and SPP in this case study were based on the Structural, Power, and Data Port from Sierra Nevada Corporation (SNC), shown in Fig. D.3 (Sullivan et al., 2018), (Sierra Nevada Corporation, 2020). This selection was made because the ports are relatively lightweight, because they enable the appropriate connections across modules, and because enough data was found on both the Active and Passive units in the SNC product catalog.

The SNC product catalog provided the mass and power data in Table D.2. The position accuracy in Table D.2 was estimated based on the SPP envelope dimensions. The components in the adjacency matrix were listed in the catalog also, but the links were created using best guesses for how the components must connect. Figure D.4 provides the adjacency matrix for the Assembly Package payload.

Figure D.3: Structural, Power, and Data Port from Sierra Nevada Corporation's product catalog (Sierra Nevada Corporation, 2020).

D.4.2 Servicing Technologies

Onboard imaging systems, image recognition software, and catalog (OIS)

The first technology needed for on-orbit servicing was the Onboard imaging systems, image recognition software, and catalog (OIS). OIS was included on the platform to locate installation locations and locate failed subsystems, so OIS supports the servicing functions of the robotic arm.

OIS was also modeled after the RPO package on the CPOD mission (Williams, 2015). Because sufficient information was already gathered for RPO, it was considered acceptable to use the same data to model a similar package of imaging hardware.

	SPA-1						SPA-2						SPP-1		SPP-2		External Links										
	Launch restraint	Electrical pass-thru harness	Visual alignment sensor	Payload locking mechanism – active	Stepper gear motor	Software_SPA	Launch restraint	Electrical pass-thru harness	Visual alignment sensor	Payload locking mechanism – active	Stepper gear motor	Software_SPA	Payload locking mechanism – passive	SPP Electrical pass-thru harness	Payload locking mechanism – passive	SPP Electrical pass-thru harness	Software_Avionics	Software_ADCS	Software_Comm	Software_Payload	Software_Propulsion	Software_Thermal	Power Control Unit	Payload locking mechanism – passive	SPP Electrical pass-thru harness	SPA Payload locking mechanism – active	SPA Electrical pass-thru harness
Launch restraint	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Electrical pass-thru harness	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13		
Visual alignment sensor	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Payload locking mechanism – active	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0		
Stepper gear motor	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Software_SPA	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0		
Launch restraint	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0		
Electrical pass-thru harness	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0		
Visual alignment sensor	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0		
Payload locking mechanism – active	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Stepper gear motor	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0		
Software_SPA	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0		
Payload locking mechanism – passive side	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			1	0
SPP Electrical pass-thru harness	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	0			0	13
Payload locking mechanism – passive side	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			0	0
SPP Electrical pass-thru harness	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			0	0
Software_Avionics	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0											
Software_ADCS	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
Software_Comm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
Software_Payload	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
Software_Propulsion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
Software_Thermal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
Power Control Unit	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0											
Payload locking mechanism – passive	0	0	0	1	0	0	0	0	0	0	0	0															
SPP Electrical pass-thru harness	0	13	0	0	0	0	0	0	0	0	0	0															
SPA Payload locking mechanism – active													1	0	0	0											
SPA Electrical pass-thru harness													0	13	0	0											

External links in same module

External links if module has no power subsystem

External links if module has power subsystem

External links to SPP of another module

External links to SPA of another module

Figure D.4: Adjacency matrix for Assembly Package payload, including 2 SPAs and 2 SPPs, with external links to other subsystems.

Unlike RPO, OIS does not include an optical target aid, and it would likely have different onboard software and algorithms. The OIS adjacency matrix and external links are displayed in Fig. D.5.

	Payload Adjacency Matrix					External Links		
	WF Imager	IR Imager	IR Imager	Optical Target Aid	OIS Processor (*Software_OIS*)	Software_ Avionics	Power Control Unit	Software _SPP
WF Imager	0	0	0	0	1	0	0	0
IR Imager	0	0	0	0	1	0	0	0
IR Imager	0	0	0	0	1	0	0	0
OIS Processor (*Software_OIS*)	1	1	1	1	0	1	0	0
Software_Avionics	0	0	0	0	1			
Power Control Unit	0	0	0	0	1			
Software_SPP	0	0	0	0	1			

External links in same module

External links if module has no power subsystem

External links if module has power subsystem

Figure D.5: Adjacency matrix for OIS payload with external links to other subsystems.

Robotic Arm for Installation (RAI)

The Robotic Arm for Installation (RAI) was designed to perform installation of modules or customer payloads. The notional docking operations are depicted in Fig. D.6, where the RAI guides modules together to dock with their respective SPA and SPP. One RAI was included on the Platform if Servicing Option 2 was selected, and one RAI was included on the Service Module in all architectures in the case study.

Recent years have seen definite progress in technology development of space robotics; however, details on the designs of robotic arms remain hard to find. Data on flight units is proprietary, and other projects are only ground demonstrations so far. Because of these limitations, the robotic arm used in this case study combines information from various sources and is largely notional.

The RAI was based on the Orbital Express Demonstration Manipulator System (OEDMS) (Ogilvie et al., 2007), (Espero, nd) and on the Front-End Robotics Enabling Near-Term Demonstration (FREND) project (Debus and Dougherty, 2009). The Orbital Express mission was a partnership between DARPA, Boeing, and MDA Corporation and was credited with demonstrating the feasibility of autonomous on-

orbit servicing after its flight in 2007. Mass and power data found online for the OEDMS (Espero, nd) was used in Table D.5 for the RAI. The position accuracy that the RAI could provide was estimated based on servo trajectory tracking performance from the OEDMS vision system during its flight (Ogilvie et al., 2007).

The FREND concept was sponsored by DARPA to grapple and service spacecraft not originally designed for servicing (Debus and Dougherty, 2009). The components and connectivity of the RAI was based on publicly available reports on FREND. Components include: two arm sections plus an end effector after the wrist, joints, actuators, launch locks, thermal control components, and flex-harness cabling. In addition, for this case study, the RAS was designed to grapple modules using an SPA at its end effector. Therefore, the SPA mass and power were added to the robotic arm, and the SPA components were included in the adjacency matrix. The resulting adjacency matrix is provided in Fig. D.7.

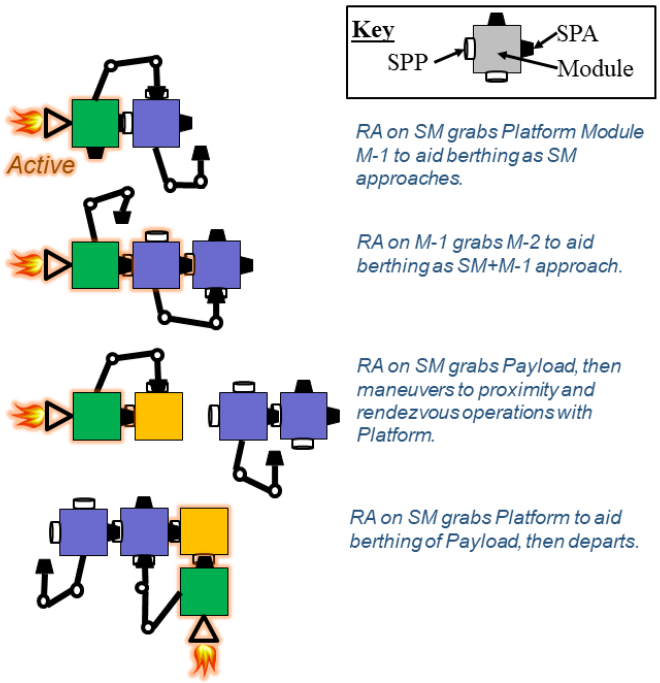


Figure D.6: Notional graphic depicting the concept of operations of an RAI to assist docking and module installation via the SPA and SPP docking ports.

	RAI																																			SPA						External Links		
	Upper arm	Forearm	Joint – shoulder	Joint – elbow	Joint – wrist1	Joint – wrist2	Joint – wrist3	Lock – shoulder yaw	Lock – elbow	Lock – wrist pitch	Lock – tool drive	Pin-puller 1	Pin-puller 2	Pin-puller 3	Pin-puller 4	Actuator 1	Actuator 2	Actuator 3	Actuator 4	Actuator 5	Actuator 6	Actuator 7	End-effector	Flex-harness cabling	Heater 1	Heater 2	Heater 3	PRT 1	PRT 2	PRT 3	Thermostat1	Thermostat2	Thermostat3	Thermal blanket	Software_RAI	Launch restraint	Electrical pass-thru harness	Visual alignment sensor	Payload locking mechanism – active	Stepper gear motor	Software_SPA	Software_Avionics	Software_SPP	Power Control Unit
Upper arm	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Forearm	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Joint – shoulder	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Joint – elbow	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
Joint – wrist1	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
Joint – wrist2	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Joint – wrist3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Lock – shoulder yaw	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Lock – elbow	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Lock – wrist pitch	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Lock – tool drive	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Pin-puller1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Pin-puller2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Pin-puller3	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Pin-puller4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Actuator 1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Actuator 2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Actuator 3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Actuator 4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Actuator 5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Actuator 6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Actuator 7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
End-effector	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Flex-harness cabling	0	0	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	0	0	0	0
Heater 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Heater 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Heater 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
PRT 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
PRT 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
PRT 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Thermostat1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Thermostat2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Thermostat3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Thermal blanket	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Software_RAI	0	0	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
Launch restraint	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Electrical pass-thru harness	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Visual alignment sensor	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Payload locking mechanism – active	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
Stepper gear motor	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0
Software_SPA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	1	0	0	0
Software_Avionics	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1										
Software_SPP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1										
Power Control Unit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1										

In same module
If module has no power subsystem
If module has power subsystem

Figure D.7: Adjacency matrix for RAI package combined with the SPA attachment, with external links to other subsystems.

Robotic Arm for Subsystem Servicing (RAS)

The Robotic Arm for Subsystem Servicing (RAS) was very similar to the RAI. However, the RAS represented a higher level of servicing capability, since it was de-

	RAS																																								SPA						External Links		
	Upper arm	Forearm	Joint – shoulder	Joint – elbow	Joint – wrist 1	Joint – wrist 2	Joint – wrist 3	Lock – shoulder yaw	Lock – elbow	Lock – wrist pitch	Lock – tool drive	Pin-puller 1	Pin-puller 2	Pin-puller 3	Pin-puller 4	Actuator 1	Actuator 2	Actuator 3	Actuator 4	Actuator 5	Actuator 6	Actuator 7	End-effector 1	End-effector 2	Flex-harness cabling	Heater 1	Heater 2	Heater 3	PRT 1	PRT 2	PRT 3	Thermostat 1	Thermostat 2	Thermostat 3	Thermal blanket	Software_RAS	Launch restraint	Electrical pass-thru harness	Visual alignment sensor	Payload locking mechanism – active	Stepper gear motor	Software_SPA	Software_Avionics	Power Control Unit	Software_SPP
Upper arm	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Forearm	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Joint – shoulder	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Joint – elbow	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Joint – wrist 1	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Joint – wrist 2	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Joint – wrist 3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Lock – shoulder yaw	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Lock – elbow	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Lock – wrist pitch	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Lock – tool drive	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Pin-puller 1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Pin-puller 2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Pin-puller 3	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Pin-puller 4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Actuator 1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Actuator 2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Actuator 3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Actuator 4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Actuator 5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Actuator 6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Actuator 7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
End-effector 1	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
End-effector 2	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Flex-harness cabling	0	0	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	0	1	0	0	0	1	0	0	0
Heater 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Heater 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Heater 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
PRT 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
PRT 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
PRT 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Thermostat 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Thermostat 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Thermostat 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Thermal blanket	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Software_RAS	0	0	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
Launch restraint	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1			
Electrical pass-thru harness	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1			
Visual alignment sensor	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1			
Payload locking mechanism – active	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0				
Stepper gear motor	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0				
Software_SPA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0			
Software_Avionics	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1									
Software_SPP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1									
Power Control Unit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1									

In same module
If module has no power subsystem
If module has power subsystem

Figure D.8: Adjacency matrix for RAS package combined with the SPA attachment, with external links to other subsystems.

signed to remove and replace specific subsystems, compared to full modules. Therefore, a second end effector was added to represent a more dexterous toolset for subsystem servicing. Also, the mass and power of the robotic arm was increased by 33%, based on it being a three-section arm—including upper arm, forearm, and wrist with end effector. The RAS adjacency matrix is provided in Fig. D.8.

D.4.3 Manufacturing Technologies

Laser End Effector for V&V (LVV)

In the manufacturing options, a robotic arm (RAS) was required since manufactured components would require installation. Additionally, a third end effector was required: the Laser End Effector for Verification and Validation (LVV). The LVV was designed to be a simple tool that was essentially just a laser and advanced algorithms for high fidelity, autonomous V&V. This idea is based off of Made In Space's Archinaut One concept (Made In Space, 2017). The LVV adjacency matrix is provided in Fig. D.9.

	Payload Adjacency Matrix		External Links	
	Laser	Software_LVV	Software_RAS	RAS Flex-harness cabling
Laser	0	1	0	0
Software_LVV	1	0	1	1
Software_RAS	0	1		
RAS Flex-harness cabling	0	1		

When attached to RAS

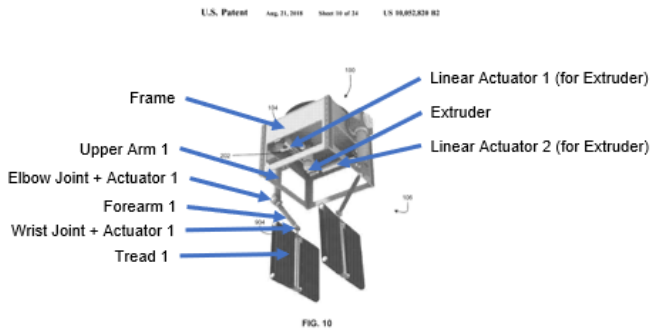
Figure D.9: Adjacency matrix for LVV payload with external links to other subsystems.

Additive Manufacturing Machine (AMM)

The Additive Manufacturing Machine (AMM) was designed to create components of an arbitrary shape and size. Depending on the feedstock provided to it, the AMM could manufacture structures, solar array panels, or electronic components. The AMM was based on the similar concept created by Made In Space, named the Extended Structures Additive Manufacturing Machine (ESAMM) (Kemmer et al., 2018) and included in the Archinaut One demonstration mission (Made In Space, 2019c). Figure D.10 provides a look at a notional ESAMM with tread arms from a patent filed by Made In Space. (Labels have been added to the figure.) The components for the AMM designed in this case study were based on the patent descriptions (Kemmer et al., 2018) and an online Archinaut One Webinar (Made In Space, 2019c); the adjacency matrix is provided in Fig. D.11.

The development of in-space manufacturing machines and 3D printers is still in nascent stages, so very little performance data was publicly available online. To create a best guess, performance parameters for various commercial 3D printers used on Earth and also the listed payload capacity of Archinaut One's spacecraft bus (Made In Space, 2019c) was compiled to determine the AMM mass and power in Table D.8.

Figure D.10: Conceptual drawing of an in-space extended structures manufacturing machine with tread arms to stabilize and maneuver the manufactured structures. Image from (Kemmer et al., 2018) with component labels added.



	Extruder	Linear actuator 1	Linear actuator 2	Upper Arm 1	Upper Arm 2	Forearm 1	Forearm 2	Tread 1	Tread 2	Joint - elbow 1	Joint - elbow 2	Joint - wrist 1	Joint - wrist 2	Joint actuator 1	Joint actuator 2	Joint actuator 3	Joint actuator 4	Frame	Software_AMM	Feedstock
Extruder	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
Linear actuator 1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Linear actuator 2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Upper Arm 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Upper Arm 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Forearm 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Forearm 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Tread 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Tread 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Joint - elbow 1	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Joint - elbow 2	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
Joint - wrist 1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0
Joint - wrist 2	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0
Joint actuator 1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
Joint actuator 2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
Joint actuator 3	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
Joint actuator 4	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
Frame	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Software_AMM	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0
Feedstock	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Figure D.11: Adjacency matrix for AMM payload with external links to other subsystems.

AMM Feedstock

Three types of manufacturing feedstock have been defined for input to the AMM: Thermoplastic Feedstock for Structures (TFS), Flexible Blanket Solar Arrays (FBS), and Feedstock for Electronics (FEL). Each of these three are added in sequence to define the three Manufacturing Options 2, 3, and 4.

The TFS was based on thermoplastic feedstock described for Made In Space's concept for Archinaut to print beams (Kugler et al., 2017). The FBS was based on

Made In Space's concept for Archinaut to produce solar panels using both printed beams and advanced flexible solar arrays (Patane et al., 2018). No information was available on the specific feedstock required to manufacture electronic components in space, although some concepts indicated that this could be feasible (Made In Space, 2017). Therefore, the FEL was purely notional. Each of these feedstock types were simple components had only one external connection—to the AMM's extruder, as seen in Fig. D.11.

Finally, the mass of manufacturing feedstock is an important consideration. Launch mass clearly can add significant cost. The feedstock mass for this case study was estimated based on predicted need for the scalability scenario:

- 40 m boom length capability from TFS
- 10 kW additional capability from FBS
- Miscellaneous electrical components from FEL (potentially wide variations in mass between different components)

Data for specific mass was compiled in Table D.10, adapted from (Kugler et al., 2017) and (Patane et al., 2018).

Feedstock	Specific Mass	Estimated Delivery Mass
(9) TFS	0.288 kg/m	11.5 kg
(10) FBS	500 W/kg	20 kg
(11) FEL	Unknown	10 kg

Table D.10: Specific mass for each type of manufacturing feedstock, which was used to estimate the mass needed in orbit with the AMM. Data adapted from (Kugler et al., 2017) and (Patane et al., 2018).