27th Annual INCOSE International Symposium (IS 2017)
Adelaide, Australia, July 15-20, 2017

Robotic Satellite Servicing Trade Space Down-Selection

Jessica Knizhnik
University of Maryland
College Park, MD 20742, USA
+1(908)591-8257
jessica.knizhnik@gmail.com

Mark Austin
University of Maryland
College Park, MD 20742, USA
+1(301)405-6627
austin@isr.umd.edu

Craig Carignan
University of Maryland
College Park, MD 20742, USA
+1(301)405-1996
craigc@ssl.umd.edu

Abstract. As remote robotic space satellite servicing technologies develop, each servicer satellite will need to account for a number of servicing scenarios and consider a variety of alternate design solutions to best meet the most servicing scenario requirements. This paper presents a graph transformation method for systematically down-selecting the number of design options available, and highlighting trade-offs in sets of design solutions which best meet satellite servicing task requirements while also reducing total mass, maximum power needed and servicing time. In the test case examined, the proposed method successfully identifies for further consideration, seven best design solutions from a set of over 100,000 potential solutions.

Introduction

Problem Statement. Government agencies and commercial entities throughout the world spend billions of dollars to send satellites to space each year. Though many of these satellites represent new science, human exploration and technology developments, many of these satellites are simply replacements for satellites that have reached the end of their lifespan. Frequently, much of the hardware aboard these satellites is still operational, but the satellite reaches the end of its lifespan due to a lack of fuel for orbital maneuvering or worn mechanisms. Rather than utilize an abundance of resources to replace a satellite entirely, it is now evident that a more cost effective solution is to simply send one additional satellite into space to robotically service a number of older, but mostly functional satellites. This strategy retains the functionality of a number of satellites for the cost of building, launching into space and operating a relatively smaller number of servicer satellites. Savings are especially likely to come from robotically servicing large fleets of satellites such as those which monitor the Earth's weather patterns to predict and follow storms, the Earth's heat signature to monitor fires and global climate or the Earth's other environmental monitoring satellites to protect humanity's home planet. Fleets of satellites are also used for commercial, military and other space telecommunications. These are considered national and international assets.

Scope and Objectives. In a step toward the application of inference-rule down-selection methods to reduce trade space options on complex systems, this paper introduces a down-selection methodology and set of graph transformations for refining a set of generic tools with a variety of specifications (descriptions of capability) in order to perform a subset of tasks needed to service a satellite. This work builds upon and is motivated by previous University of Maryland Space

Figure 1: Astronauts Servicing the Hubble Space Telescope (HST) (from the HST website)

Systems Laboratory (UMD SSL) research on the Hubble Space Telescope (HST). HST is a well-known satellite with an abundance of easily accessible data on satellite servicing. HST underwent five astronaut servicing missions during its operational life time (Figure 1).

Related Work

Robotic Servicing Satellites

Promise of Robotic Servicing Satellites. Research at NASA's Goddard Space Flight Center (GSFC) in Greenbelt, Maryland as well at the University of Maryland's (UMD) Space Systems Laboratory (SSL) in College Park, Maryland both show promise in robotic satellite servicing. Engineers at GSFC have created a high-level architecture for servicing satellites.

As illustrated in Figure 2, this architecture's hierarchy begins with a servicing mission which includes both a servicer satellite, at least one client satellite and at least one servicing task that must be performed during the servicing mission. The servicing satellite includes one robotic arm and at least one tool which can be connected to that arm via an end effector. Figure 3 shows the key interactions between each of the components in Figure 2. The tool, connected to the robotic arm, assists the servicing satellite to perform a servicing task on a client satellite. The SysML block diagram states that the servicer satellite, connects to the robotic arm, the robotic arm connects to the end effector, the end effector connects to a tool and the tool interacts with the client satellite.

There are a number of different types of tools which a robotic servicing mission could transport into space to complete its servicing task(s). The Space Applications of Automated Robotics and Machine Intelligence Systems (ARAMIS) study (Akin et. al. 1983) categorized these tools into generic categories. They include a hand, all purpose, camera/sensor, welder, cutter, latcher, gripper, bolt driver, pincher, delicate pincher computer and lubricant applicator. In addition, a safety cap remover and a fuel injector will likely also be necessary to perform refueling tasks such as those performed during GSFC's International Space Station (ISS) test Robotic Refueling Mission (RRM). Figure 4 details these generic tool types and their associated descriptive values. Multiple tools of the same type can exist and each can vary in individual specifications (listed as values in SysML). This means that in theory, there could be an infinite number of tool options and combinations of tools to use for any given servicing mission. Figure 5 shows a sample of a Wire Cutter Tool (in multiple orientations) created for RRM.

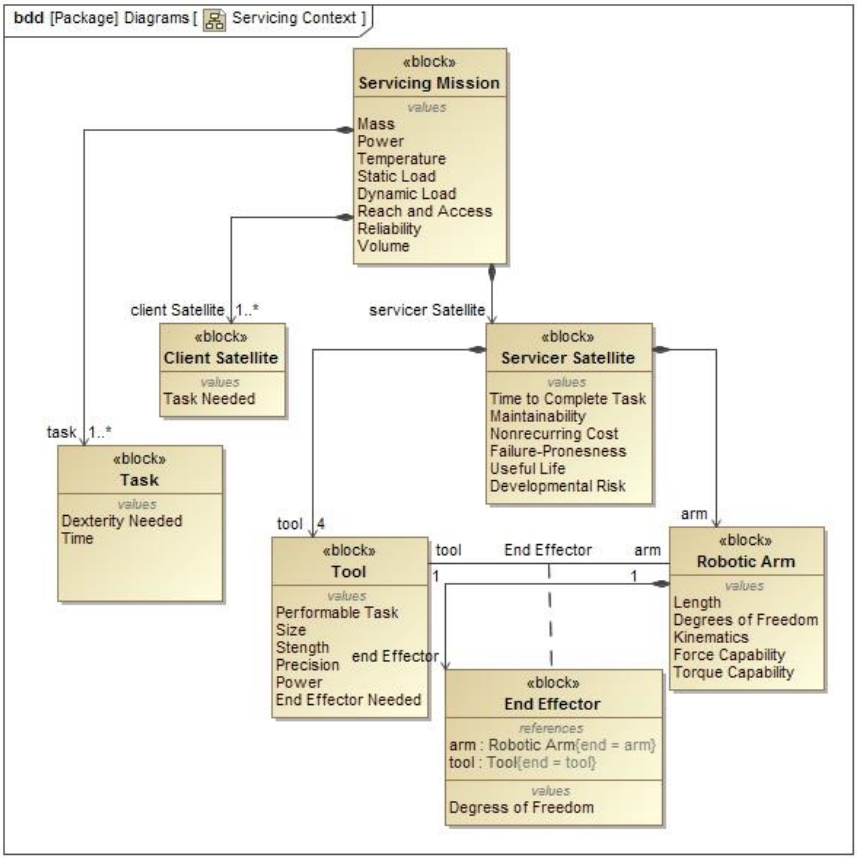

Figure 2: SysML block diagram for Satellite Servicing Architecture.

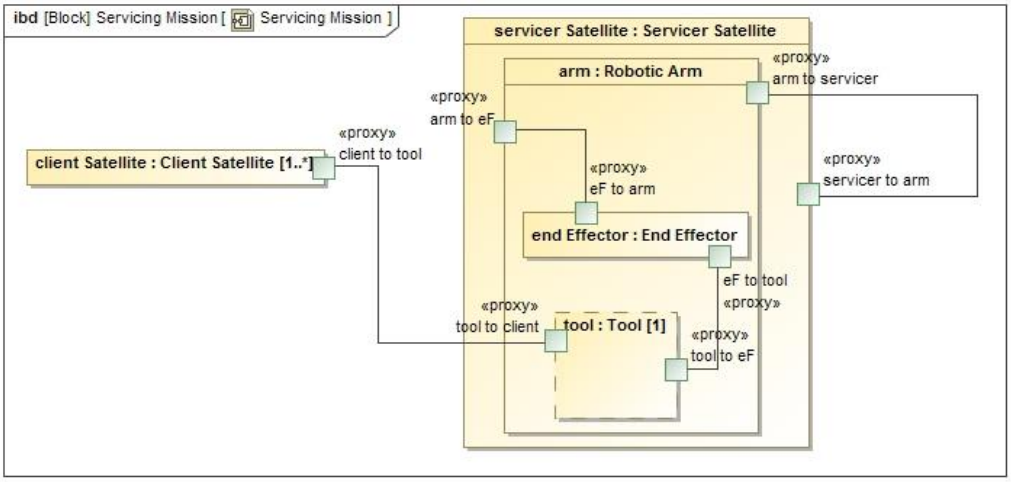

Figure 3: SysML Internal block diagram for Servicing Mission Interfaces and Interactions.

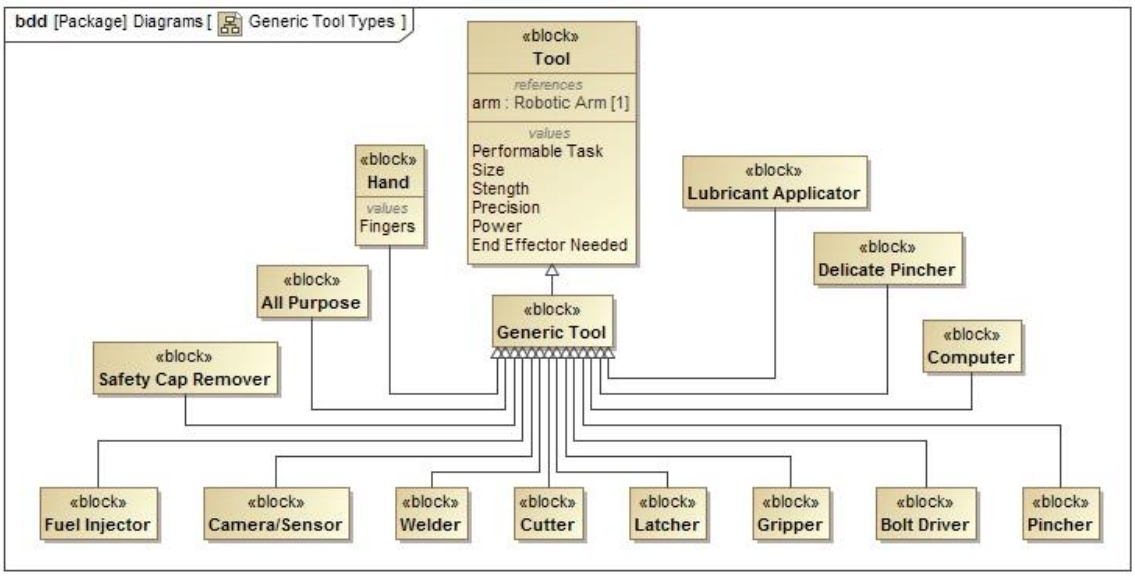

Figure 4: Generic Tool Types

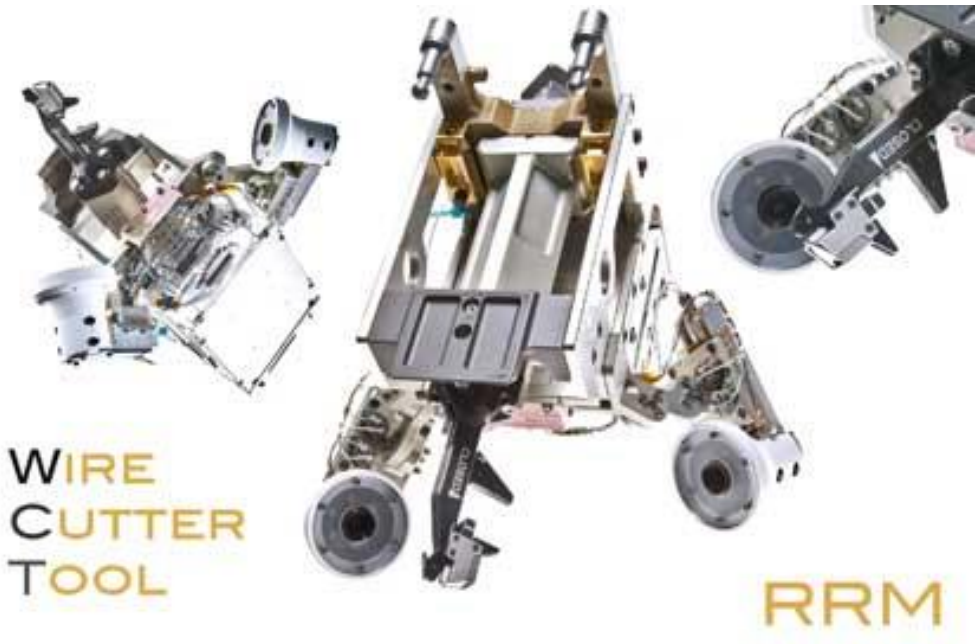

Figure 5: Robotic Refueling Mission (RRM) Wire Cutter Tool (from the RRM website)

Though there are an infinite number of tool options and tool combinations available, launch vehicles cannot lift an infinite amount of mass into space, servicer satellites cannot provide an infinite amount of power to operate these tools and client satellites cannot spend an infinite amount of nonoperational time to allow for servicing tasks to occur. For instance the GSFC's ISS RRM limited itself to a "toolbox" with four tool slots (see Figure 6).

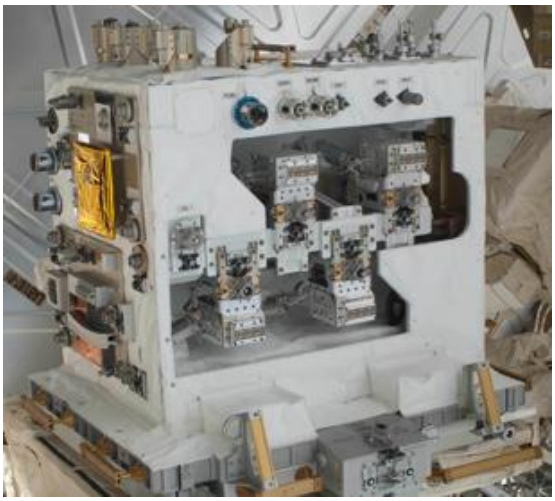

Figure 6: Robotic Refueling Mission (RRM) "Toolbox" (from the RRM website)

For this reason, it is imperative that robotic satellite servicing utilize a method for quickly and easily showing engineers their tool combinations which best meet their particular servicing mission's task requirements while also reducing tool mass, power and task time.

Trade-Space Down Selection

Previous Work. Researchers (Nassar and Austin, 2013) at the Institute for Systems Research (ISR) at the University of Maryland, College Park have designed computational procedures for the systematic transformation of user requirements, high-level models of system architecture, and libraries of components into collections of viable design alternatives supported by trade-spaces for deign consideration. These procedures fall into a general class of problems called the component selection problems (see Figure 7).

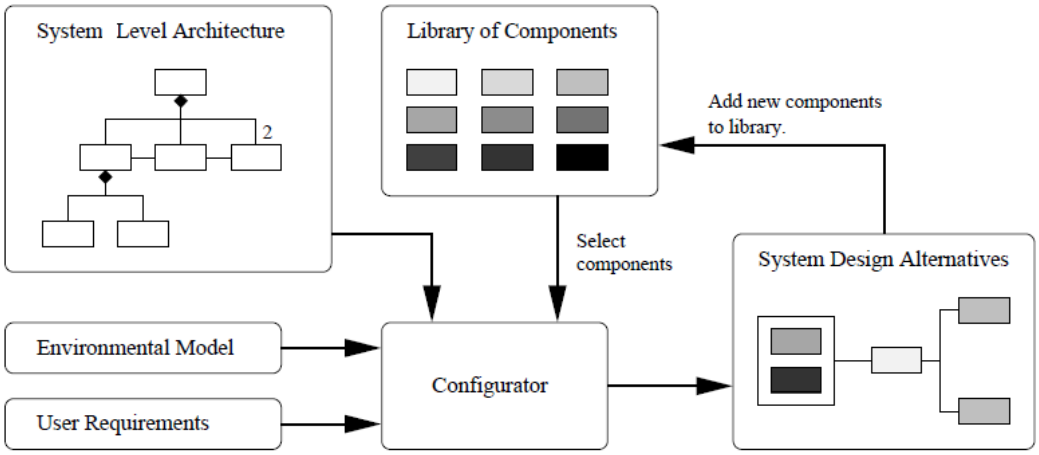

Figure 7: Component-Selection Design Problem (Adapted from Nassar and Austin (2013)).

Figure 8 shows the step-by-step procedure for the application of inference mechanisms on graph transformations. Notice that compatibility (or lack thereof) relations between sets of components are evaluated before the problem requirements are considered. One can think of these procedures as ``computational sculpting'' where sets of design alternatives and the associated trade space curves are created through the systematic application of inference-guided transformations on graphs. Nassar

and Austin (2013) demonstrated this approach on a problem that involved selection of components from a library for a home theater system. The requirements, components, and system architecture were all modeled as collections of resource description framework (RDF) graphs. RDF provides a general means for representing graphs of resources on the Web and, as such, is an ideal way to represent heterogeneous data in design. The ensuing inference procedures and graph transformations that work toward feasible design solutions were implemented in Python. The work in this paper serves to take the next step towards this goal by applying inference-rule down selection methods to more complex, space based applications.

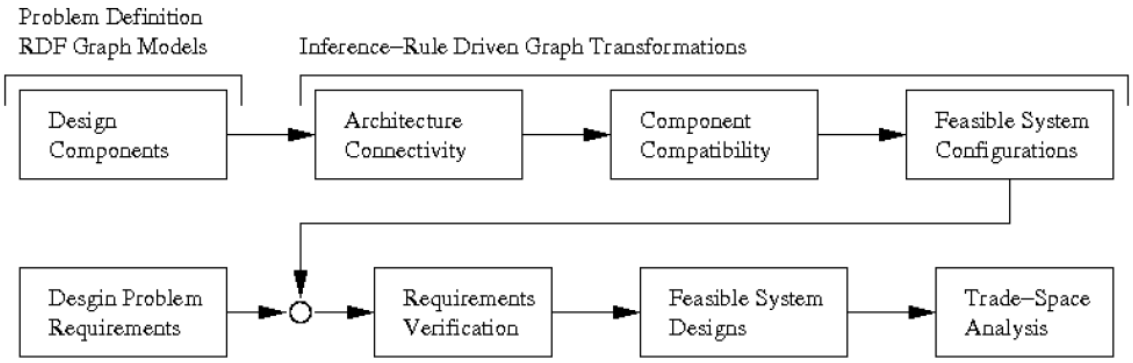

Figure 8: Nassar's and Austin's (2013) Flowchart of Activities for Problem Definition with RDF Graph Models Followed by Inference-Rule Driven Graph Transformations

The RDF/Python approach to implementation is not the only pathway forward. For example, the same approach could involve Web Ontology Language (OWL) technologies, Jena graphs and Jena Rules. This is a step that is yet to be explored. Another possibility is to code the component selection problem as a mixed-integer programming problem and compute solutions in a commercial optimization package such as CPLEX. We note, however, that a key advantage of the proposed approach is the explicit representation and application of rules which enhance understanding for how the system design alternatives and trade-space curves are being generated.

Methodology Demonstration

Trade-Space Down Selection for Servicing Spacecraft. We propose a down selection methodology and set of graph transformations for refining a set of generic tools with varying specification in order to perform a subset of tasks needed to service satellite operations. The case study application is servicing of the Hubble Spacecraft (HST).

Figure 9 shows a step-by-step procedure for generating a manageable set of viable tool combination solutions. The key points are as follows:

- Steps 1, 2 and 4 input the necessary tools and constraints needed to perform the down-selection. Steps 1 and 2 comprise the "Design Components" block from the inference-rule down selection process described in Figure 8. Step 4 is the "Design Problem Requirements" block. Step 12 outputs the final design space and shows the engineer all viable design solutions as well as those which are most optimal from the remaining solution set. This is the "Trade Space Analysis" block in Figure 8. In between these steps, the algorithm conducts a series of graph transformations.

- Steps 5, 7, 8, 9 and 11 all reorganize the design options to allow for requirement and constraint application. Steps 5, 7, 8 and 9 are all part of the "Architecture Connectivity" block in Figure 8. Step 11 is the "Feasible System Configurations" and the "Feasible System Designs" blocks.

- Steps 3, 6 and 10 all remove design solutions which do not meet system constraints. Step 3 is the "Component Compatibility" block in Figure 8. Steps 6 and 10 are both the "Requirements Verification" block.

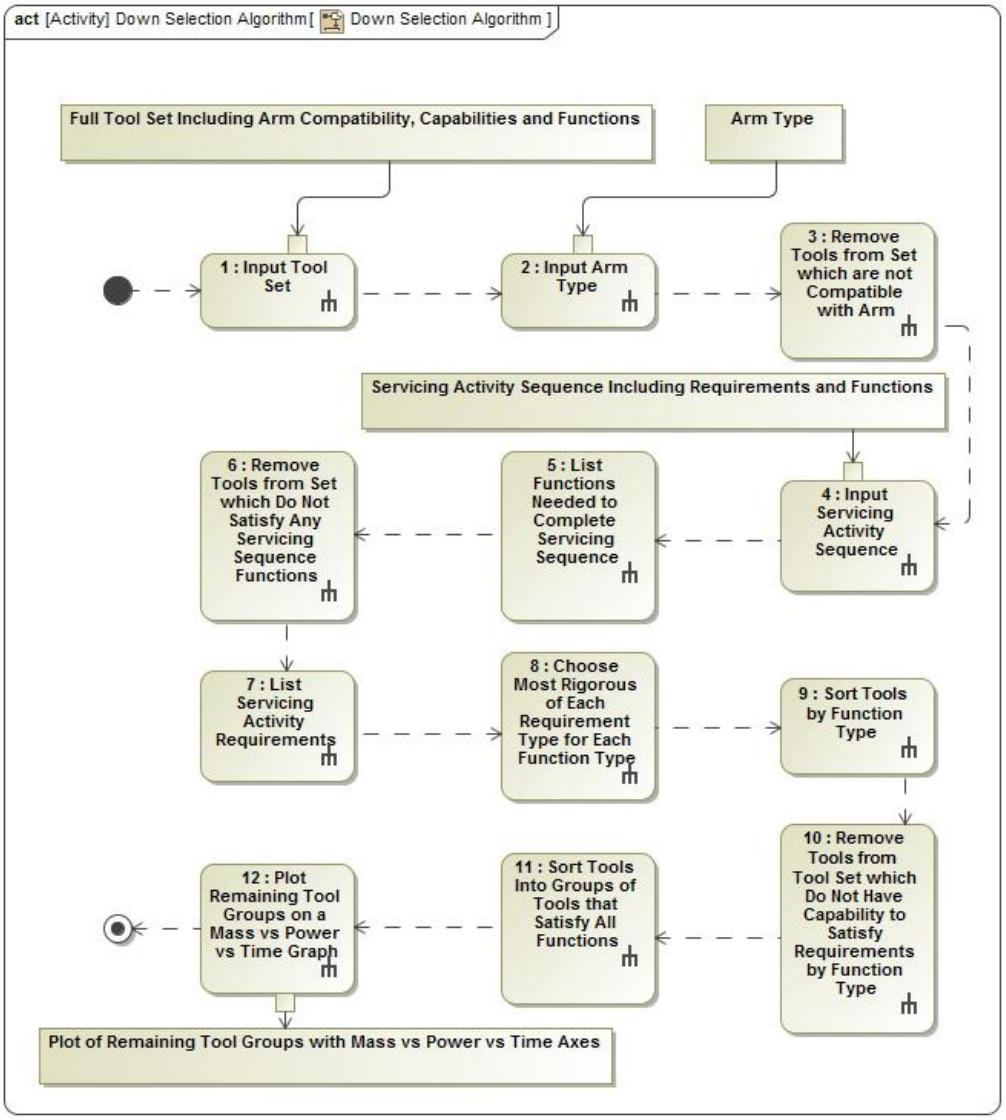

Figure 9: Down-Selection Algorithm

System Modeling of the Hubble Spacecraft. The HST is made up of a large cylindrical spacecraft with two solar arrays attached on either end. The spacecraft is comprised of its external structure as well as a suite of science instruments, an Optical Telescope Assembly (OTA) and a support system. The science instruments, OTA, support system and solar arrays all connect to the spacecraft via its structure. Each of these systems have subsystems and components (such as the instruments, mirrors, reaction wheels, etc.) which have been serviced during one of the five astronaut servicing missions.

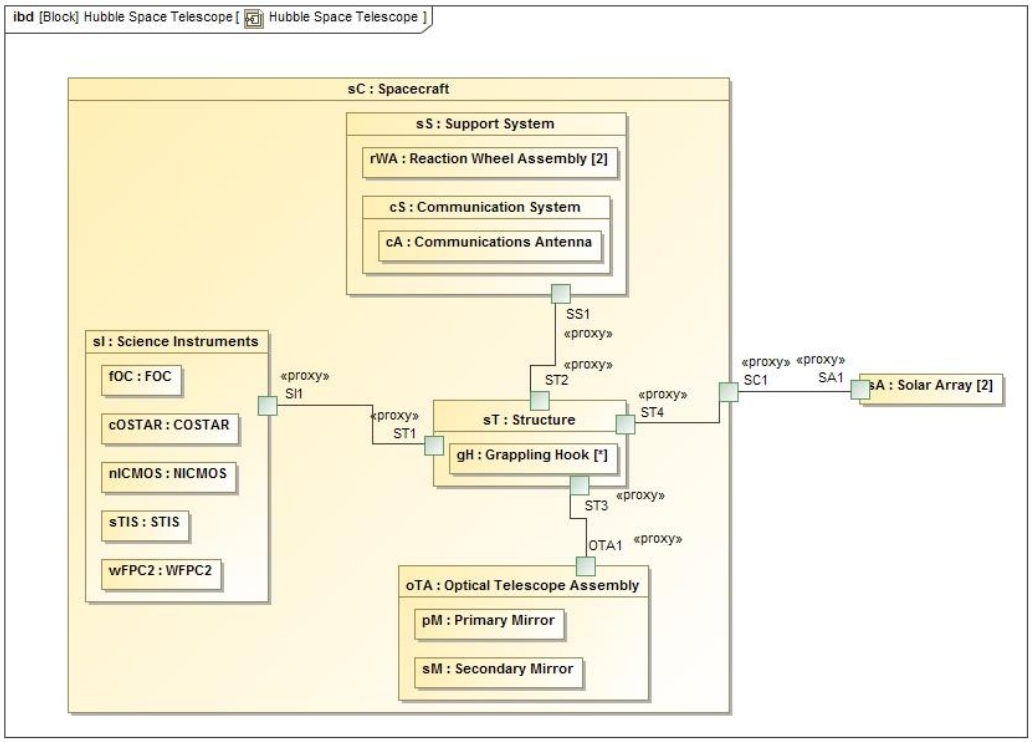

Figure 10: Hubble Space Telescope (HST) System Architecture and Interfaces (as defined on the HST website)

Video and Photographic Footage of Servicing Operations. Pilotte (2004) utilized HST astronaut Servicing Mission 3B (SM3B) as a basis for studying methods for robotically servicing satellites. The study reviewed hours of video and photographic footage taken during the Extra Vehicular Activities (EVAs) performed during that mission in order to create a table of tasks and subtasks executed during SM3B along with the likely robotic servicing tools necessary to complete each task and sub task activity. A portion of this table is shown in Table 1.

Table 1: Selection of Tools Needed for Hubble Space Telescope (HST) Robotic Servicing from Pilotte (2004)

Ref #	EV	Primitive	Task Name	Need?	Broad Prim	1st EE	Inst #	2nd EE	Inst #
3|11		**PCU-R Mate**							
3226	R MS	Mate connectors (2-bottom PCU-R)	PCU-R Mate	Yes	mate/demate PCU connector	Pinch on retainer	2	HT Pinch	2
3227	R MS	Mate connectors (34-left PCU-R)	PCU-R Mate	Yes	mate/demate PCU connector	Pinch on retainer	34	HT Pinch	34
3228	R MS	Stow J13/J14 saver caps in trash bag	PCU-R Mate	Yes	stow connector cap	Pinch	2		
3|12		**V2 Aft Shroud Handrail Covers**							
3230	FF	inspect +/- V2 handrails used for ACS and NCS	V2 Aft Shroud Handrail Covers	Yes	inspect worksite	Camera	1		
3231	FF	retrieve handrail covers from ASIPE	V2 Aft Shroud Handrail Covers	Yes	retrieve handrail covers	Unknown	1		
3232	FF	install handrail covers	V2 Aft Shroud Handrail Covers	Yes	install handrail covers	Unknown	1		
3233	FF	config. HST PFR (aft ASIPE) for ACS	V2 Aft Shroud Handrail Covers	No		Unknown	1		

Each activity in the table has an associated reference number ("Ref #"), initials of the Extra Vehicular astronaut who originally performed the task ("EV")), name ("Primitive"), larger task it assists in completing ("Task Name"), information on its necessity for completing the servicing scenario ("Need?"), a general categorization ("Broad Prim"), the first tool needed ("1st EE"), the number of times the first tool is needed ("Inst #"), the second tool needed ("2nd EE") and the number of times the second tool is needed.

Demonstrating the Down-Selection Methodology. Table 2 shows the initial 19 servicing tool options used to demonstrate this algorithm (Figure 9, Step 1). Each tool option can be used for either the RESTORE arm type (the arm to be used in NASA's RESTORE-L servicing mission) or the DEXTRE arm type (the arm used on RRM). After choosing the RESTORE arm type for this demonstration (Figure 9, Step 2), tools 4, 7, 13, 14 and 19 were all removed from the set of tool options (Figure 9, Step 3 is shown in red in Table 2).

Table 2: Initial Set of Tool Options

ID	Tool	Functions	Arm	Force	Resolution	Size	Step Removed
1	Delicate Pinch	Delicate Pinch	RESTORE	1		10	
2	Delicate Pinch	Delicate Pinch	RESTORE	2		9	
3	Delicate Pinch	Delicate Pinch	RESTORE	20		15	10
4	Delicate Pinch	Delicate Pinch	DEXTRE	10		20	3
5	Welder	Welder	RESTORE	5		12	6
6	Cutter	Cutter	RESTORE	13		13	6
7	Pinch	Pinch	DEXTRE	1		8	3
8	Pinch	Pinch	RESTORE	6		12	
9	Pinch	Pinch	RESTORE	7		13	
10	Bolt Driver	Bolt Driver	RESTORE	5		18	
11	Bolt Driver	Bolt Driver	RESTORE	4		30	6
12	Multi Tool	Delicate Pinch, Pinch and Camera	RESTORE	5	22	10	
13	Grip	Grip	DEXTRE	1		1	3
14	Grip	Grip	DEXTRE	2		2	3
15	Grip	Grip	RESTORE	5		5	10
16	Grip	Grip	RESTORE	20		4	
17	Camera	Camera	RESTORE	0	30	5	
18	Camera	Camera	RESTORE	0	21	11	10
19	Camera	Camera	DEXTRE	0	20	10	3

Figure 9, Step 4, then calls to import a servicing activity sequence along with an associated set of requirements and functions for that sequence. This information is shown in Table 3. These activities are all sample activities from the Pilotte (2004) work. The force, resolution and size requirements and specifications listed in Table 2 and Table 3 respectively were arbitrarily generated without units for demonstration purposes only.

Table 3: Servicing Activity Sequence and Associated Requirements

ID	Activity	Tool Function	Force	Resolution	Size
1	Stow groundstrap (SA-3)	Delicate pinch	<5		<10
2	Remove PIP pin (fwd latch)	PIP (pinch)	4<x<10		9<x<20
3	Remove BAPS post	Small handrail (grip)	>10		<5

4	Inspect p105 and p106 covers	Camera		>20	<10

Figure 9, Step 5 and Step 6 next call to list the functions needed to complete the servicing activity sequence, as done in Table 3, and remove tools from the tool set which do not satisfy these requirements, as shown in orange in Table 2. Because each tool function is only listed once, the requirements listed (Figure 9, Step 7) in Table 3 for each tool function are the most rigorous available by default (satisfying Figure 9, Step 8). Table 2 has already been configured to show tools by function type (Figure 9, Step 9) and shows tools removed which do not meet any of the Table 3 requirements in a yellow color (Figure 9, Step 10).

Next, the algorithm calls to organize the remaining tools into sets of tools which satisfy all of the servicing activity functions needed in Table 3. Table 4 shows all 18 viable tool combination groups (satisfying Figure 9, Step 11).

Table 4: Viable Tool Combination Groups

Group ID	Tool IDs
1	1, 8, 12, 16
2	1, 9, 12, 16
3	1, 8, 16, 17
4	1, 9, 16, 17
5	2, 8, 12, 16
6	2, 9, 12, 16
7	2, 8, 16, 17
8	2, 9, 16, 17
9	8, 12, 16
10	9, 12, 16
11	1, 12, 16
12	2, 12, 16
13	12, 16
14	8, 12, 16, 17
15	9, 12, 16, 17
16	1, 12, 16, 17
17	2, 12, 16, 17
18	12, 16, 17

Generation of Trade-off Curves. Finally, the algorithm generates tradeoff curves for tool groups versus tool group mass, total task time and maximum tool power needed (in satisfaction of Figure 9, Step 11). Figure 11, Figure 12 and Figure 13 show these plots and highlight the tool groups which minimize mass, power or time, in red triangles from the remaining tools in Table 2's initial tool set. Table 5 shows the tools' individual specifications for reference.

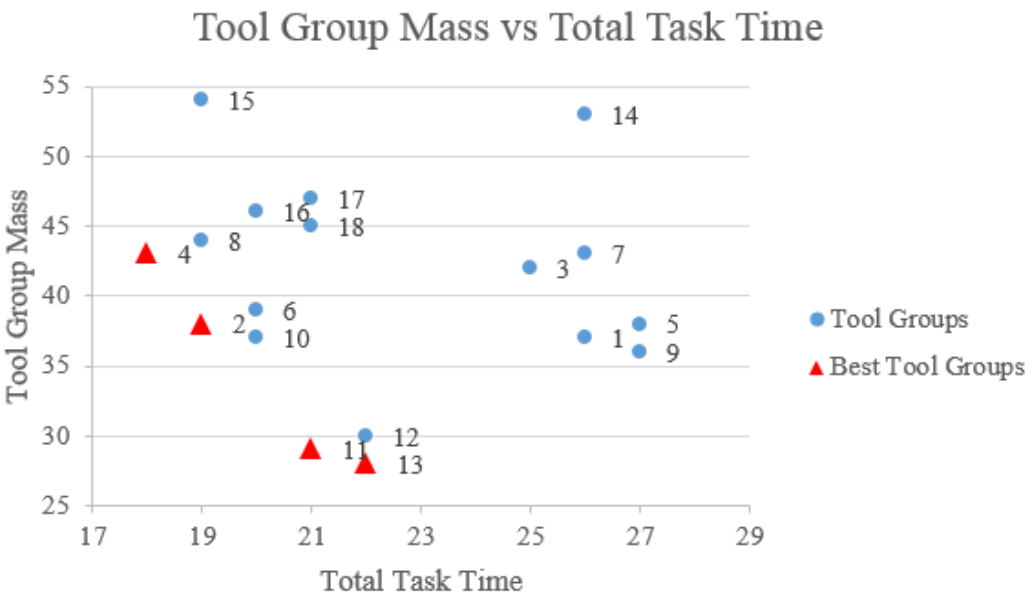

Figure 11: Tool Group Mass vs Total Task Time

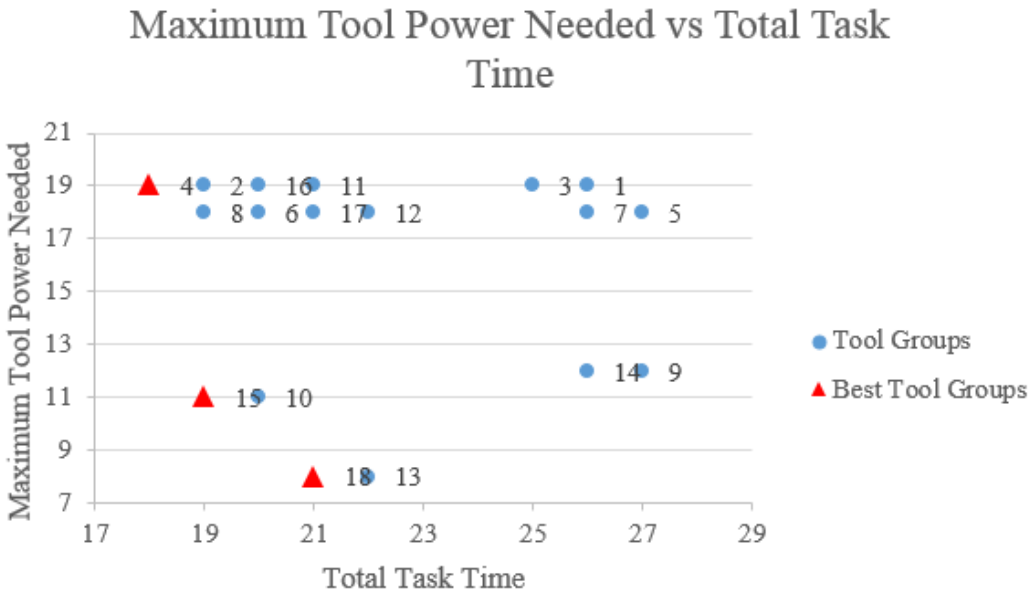

Figure 12: Maximum Tool Power Needed vs Total Task Time

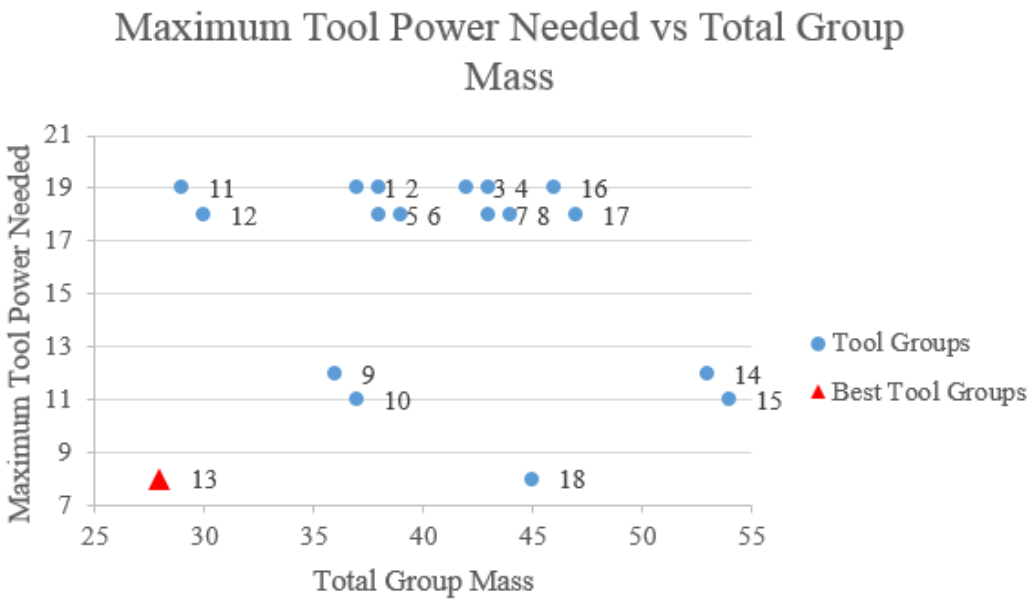

Figure 13: Maximum Tool Power Need vs Total Group Mass

Table 5: Viable Tool Specifications

ID	Tool	Functions	Mass	Power	Time to Complete Task 1	Time to Complete Task 2	Time to Complete Task 3	Time to Complete Task 4
1	Delicate Pinch	Delicate Pinch	1	19	1			
2	Delicate Pinch	Delicate Pinch	2	18	2			
8	Pinch	Pinch	8	12		8		
9	Pinch	Pinch	9	11		1		
12	Multi Tool	Delicate Pinch, Pinch and Camera	12	8	2	3		7
16	Grip	Grip	16	4			10	
17	Camera	Camera	17	3				6

In short, after listing all available tools and applying inference-rules (Table 2), the down-selection method whittled the trade space down to a set of 18 viable tool combinations. These tool combinations were then easily compared with each other on the bases of mass, power and time.

Interpretation. From this point, an engineer could note which tool group are most optimal in each plot and determine which tool group to choose as the design solution. In this particular case, because tool group 13 is the most optimal in Figure 11 and Figure 13 as well as near optimal in Figure 12, it is likely the best design solution to this sample tool trade study. Thus, rather than looking at an overwhelmingly large selection of choices, the engineer has a much smaller and much more manageable decision available without expending the resources necessary to meet optimization algorithm conditions or to run more computationally complex optimization algorithms on all of the possible tools and tool combinations available.

Conclusions and Future Work

Conclusions. The down-selection method presented in this study has tremendous potential for efficiently and manageably identifying a best design solutions from a large option set. The method also has the potential to showcase system properties to engineers that they may not have deduced without performing a trade study. In this scenario, even though the multi tool may have been one of the most massive tools with mediocre power consumption and task completion times, it still provides an overall savings in these areas when utilizing all of its functions (tool group 13). However, only when comparing tool group options against mass and power did it become evident that tool group 13 the clear best choice. If engineers do not need to consider mass, then groups 4, 15 and 18 all present viable design options. Of those options, only one of them uses three instead of four tools (group 18) and one of those options does not include the multi tool (group 4). The down-selection method presented in this study identified cases when the multi tool is and is not most appropriate.

Globally, the graph transformation method for systematic trade space down-selection presented in this study successfully presented a set of 18 servicing tool group options for a set of four servicing tasks from an initial set of 19 potential servicing tools, amounting to an initial 19^4=130,321 potential servicing tool group combinations. Of those 18 servicing tool groups, an engineer could then easily visualize the seven best group options as measured against mass, power and time and use engineering judgement to pick the single ultimate design solution. This systematic trade space down-selection method shows promise for quickly reducing an even larger, more intractable problem to one that is easily solvable.

Future Work. Because this graph transformation method for systematic trade space down-selection successfully reduced the trade space from over 100,000 options to 18 options with an even smaller number of clear winning options, the method has the potential to reduce an even larger set of potential design solutions to a smaller set of easily comparable set of solutions. To do this, the down selection algorithm must be automated. Future work will create an executable software program which can run through orders of magnitude more design options for the satellite servicing scenario. Such a program would ideally be linked with common commercial tools such as MagicDraw which can describe systems with the SysML standard. This program could include methods for accounting for specification units so that engineers need not standardize units before inputting them. In addition, though the algorithm presented in this study assumes that a robotic arm has already been chosen, future iterations of the algorithm could include different arm options within the trade space. This is an exciting and undoubtedly fruitful avenue for improving satellite servicing.

Scaling the algorithm up to accommodate a larger set of initial tool options and requirements will require automation. Automatically characterizing and evaluating the trade space may prove challenging because it must be done in a rigorous ontological fashion. Delgoshaei and Austin (2012) and Mosteller et. al. (2012) have performed similar work for transit system and biomedical system applications respectively. Hennig et. al. (2011) have also studied space system ontologies, though not for the robotic servicing application or for direct use in automated algorithms. We will explore avenues for casting down selection processes as a finite domain constraint programming problem, and then employing a variety of strategies (algorithms) to systematically prune the design space. A second possible approach is to employ Semantic Web technologies for the graph representation of relationships among various design options – in this case, arms and tools – and use sets of rules to systematically prune the semantic graph until all of the design alternatives are feasible and can be ranked and visualized in trade-off diagrams. To improve the accuracy with which requirements are expressed and evaluated, there is a strong need for computational procedures and tools that can work with notions of time, space, currency and other units of measure, and incorporated them into Boolean, equality, and inequality constraints. The automated algorithm will likely build upon work that Delgoshaei and Petnga have recently completed (Petnga, 2016; Delgoshaei, 2014).

Future iterations of the algorithm presented in this study could also include arm selection within the trade space, however this could also prove difficult to execute because robotic arm requirements depend on complex physics. Simple inequalities may not accurately represent these requirements like they do servicing tool requirements. The algorithm will need to only include the physics that is most important to robotic arm selection in order to simplify computation.

References

Akin, D. L., Minsky, M. L., Thiel, E. D. and Kurtzman, C. R., 1983, 'Space Applications of Automation, Robotics and Machine Intelligence Systems (ARAMIS) – Phase II, Volume 1: Telepresence Technology Base Development', NASA Contractor Report 3734, Massachusetts Institute of Technology, Cambridge, MA, US.

Delgoshaei, P. and Austin, M., 2012, 'Software Patterns for Traceability of Requirements to Finite State Machine Behavior: Application to Rail Transit Systems Design and Management', INCOSE International Symposium Paper, University of Maryland, College Park, MD, US.

Delgoshaei P., Austin, M., and Pertzborn A., A Semantic Framework for Modeling and Simulation of Cyber-Physical Systems, International Journal On Advances in Systems and Measurements, Vol. 7, No's 3-4, December 2014.

Hennig, C., Viehl, A., Kämpgen, B. and Eisenmann, H., 2011, 'Ontology-Based Design of Space Systems', Airbus Defence and Space, Space Systems, Friedrichshafen, Germany and FZI Research Center for Information Technology, Karlsruhe, Germany

Hubble Space Telescope. See: http://hubblesite.org/, 2016.

Mosteller, M., Austin, M., Yang S., and Ghodssi R., "Platforms for Engineering Experimen- tal Biomedical Systems," 22nd Annual International Symposium of The International Council on Systems Engineering (INCOSE 2012), Rome, Italy, 2012.

Nassar, N. and Austin, M., 2013, 'Model-Based Systems Engineering Design and Trade-Off Analysis with RDF Graphs', Conference on Systems Engineering Research, Atlanta, GA, US.

Petgna, L., Austin, M., and Blackburn M., "Semantically-enabled Model-based Systems Engineering of Safety-Critical Network of Systems," INCOSE 2017, Adelaide, Australia, July 15 – 20, 2017.

Pilotte, K. J., 2004, 'Analysis of Grasp Requirements for Telerobotic Satellite Servicing', Master of Science Thesis, University of Maryland, College Park, MD, US.

RESTORE-L Mission. See: http://www.nasa.gov/feature/nasa-s-restore-l-mission-to-refuel-landsat-7-demonstrate-crosscutting-technologies, 2016.

Robotic Refueling Mission. See: https://ssco.gsfc.nasa.gov/rrm_tools.html and https://ssco.gsfc.nasa.gov/rrm_tasks.html, 2016.

Biography

Jessica Knizhnik. Jessica Knizhnik is currently completing her MS in Systems Engineering at the University of Maryland where her research focuses on using model based techniques for exploring and automating complex satellite design trade studies. When not working towards her MS degree, she works for NASA as Co-Lead for its MBSE Pathfinder effort to understand MBSE implementation by utilizing test cases and engineering expertise from across all ten NASA field centers in concert with partners in industry and academia. Prior to her MS work, she received her BS in Mechanical Engineering from the University of Maryland.

Mark Austin. Mark Austin is an Associate Professor of Civil and Environmental Engineering at the University of Maryland, College Park, with a joint appointment in the Institute for Systems Research (ISR). Mark has served as Technical Director of the Master of Science in Systems Engineering (MSSE) Program at ISR. Mark has a Bachelor of Civil Engineering (First Class Honors) from the

University of Canterbury, Christchurch, New Zealand, and M.S. and Ph.D. degrees in Structural Engineering from the University of California, Berkeley.

Craig Carignan. Craig Carignan received his Ph.D. in robotics from the Massachusetts Institute of Technology in 1987, and afterwards worked at the NASA Goddard Space Flight Center developing and testing control systems for a robot flight experiment. He served as a research associate at the University of Maryland Space Systems Laboratory developing control systems for underwater robotic vehicles, and launched a rehabilitation robotics effort at Georgetown University in 2006. He currently serves as an adjunct faculty at the University of Maryland and is currently developing robot control software for a NASA satellite servicing experiment to be launched in 2019.