Space Systems, Policy, and Architecture Research Consortium (SSPARC)
Draft 10/5/04

"SSPARC BOOK" MATERIAL for Lecture 5

Prepared by:

Hugh McManus
Metis Design

Draft for evaluation only. Do not distribute.

TABLE OF CONTENTS

DEDICATION AND NOTE ON SOURCES 3
8. MODELING 4
    8.1. Approach 4
        Organization of data 4
    8.2. Creating models 5
        Organizing models – the Design Structure Matrix (DSM) 6
        A note on iteration and optimizations 6
        Tools for creating models 7
        A note on multi-concept modeling 9
    8.3. X-TOS Example 10
    8.4. SPACETUG Example 13
9. EXPLORING THE TRADESPACE 15
    9.1. Goal of Tradespace Exploration 15
    9.2. Techniques: Be Curious! 16
        The Big Picture 17
        The Pareto Front 18
        Parametric and Sensitivity Studies 18
        Looking at Designs in Depth 19
    9.3. X-TOS Tradespace 19
        Re-evaluation of the tradespace with changing user preferences 23
    9.4. Spacetug Tradespace 25
    9.5. Further Examples of Tradespace Exploration 30
        Terrestrial Planet Finder 30
        Broadband 32
        A-TOS 33
        B-TOS 34
NOTES AND REFERENCES 35

DEDICATION AND NOTE ON SOURCES

This document is an excerpt of a future book or hyper-book on the MATE-CON method. It is provided for class use as a draft. Suggestions for improvement are welcome, as are warnings of errors or omissions. The notes below apply to the entire work in progress; the work or excerpts of it should not be reproduced in any form without these notes.

This document is dedicated to the memory of Joyce Warmkessel, a colleague, mentor, and friend to many in the SSPARC and LAI communities. Many of the core ideas behind this work were originally expressed and developed by her, and she was a key mentor and facilitator to the development of all of this work.

The content of this document was developed by the SSPARC consortium. The primary compilers and codifiers of the MATE-CON method were Lt. Nathan Dillar and Adam Ross, in Master's thesis entitled, respectively, "Utilizing Multiple Attribute Tradespace Exploration with Concurrent Design for Creating Aerospace Systems Requirement,"[1] and "Multi-Attribute Tradespace Exploration with Concurrent Design as a Value-Centric Framework for Space System Architecture and Design."[2] Major contributors of the original concepts within the method, and/or complimentary methods and tools, include our SSPARC faculty and staff colleagues Elisabeth Paté-Cornell of Stanford University, Joel Sercel and Fred Cullick of Cal Tech, and Amar Gupta of MIT, post-doctoral researcher Bill Kaliardos, and graduate students Jimmy Benjamin, Jason Derleth, Bobak Ferdowsi, Dave Ferris, Russ Garber, Andre Girerd, Seth Guikema, Cyrus Jilla, Chris Roberts, Satwik Seshasai, Nirav Shah, Todd Shuman, Tim Spaulding, Dave Stagney, Dan Thunnissen, Myles Walton, Annalisa Wiegel, and Brandon Wood, along with their advisors and committees. Many other students, staff, and undergraduate researchers also contributed. Bill Borer, Kevin Ray, and John Ballenthin of the Air Force Research Laboratory, Steve Wall of NASA JPL, and Pete Hendrickson of the Department of Defense aided with the development of the method and the development of the case studies. SSPARC research work has been supported by an active group of industry practitioners, through both an Industrial Advisory Board (IAB) and on-site implementation activities.

The text of this manual is built on SSPARC research and member documents. Much of its contents are excerpts, modifications, or paraphrases of published or unpublished work done under SSPARC sponsorship. Every effort has been made to correctly attribute all contributions. Word-for-word excerpts are identified with quotes or indented, with citations. Many other excerpts have been edited to varying degrees and are integrated into the text for clarity. Their sources are cited in the text or in endnotes. Any omissions or errors of attribution should be brought to the authors' immediate attention for correction.

8. MODELING[3]

In previous weeks, we have scoped a problem, defined the attributes that a system would have to have to address it, quantified the utilities of the attributes, and defined a solution concepts and a design vector that contains many possible versions of that concept. It is now necessary to tie these together with models.

In this section, we will not attempt to either cover the large field of multi-disciplinary modeling, or provide a proscriptive process for creating a model for a MATE analysis. In stead, a modeling approach is suggested that allows a traceable calculation of attributes, utilities, and costs given an enumerated vector of possible designs or architectures. The approach allows the models to be adaptable to the changing needs of an evolving tradespace analysis. Some tools for organizing and carrying out the modeling are mentioned. None are required to do a MATE analysis, and none are covered in any great depth.

8.1.Approach

The basic architecture of a MATE modeling effort is shown in Figure 8-1. The design and constants vectors are inputs to a model of the system. For each set of values in the design vector, representing a single candidate architecture or design, the model calculates the desired attributes. The attributes in turn determine the utility of this candidate. Some important quantities are also extracted from the model; these are referred to as intermediate variables. The attributes and some of the intermediate variables are used as inputs to a cost model, and the cost of the candidate is calculated. This processes is repeated for all (or a selected subset) of the sets of values in the design vector, resulting in a populated utility vs. cost tradespace.

Note that to fully enumerate the tradespace, these models must be run many times, so there is a premium on creating models with the right degree of fidelity, and on computational efficiency of the integrated model set.

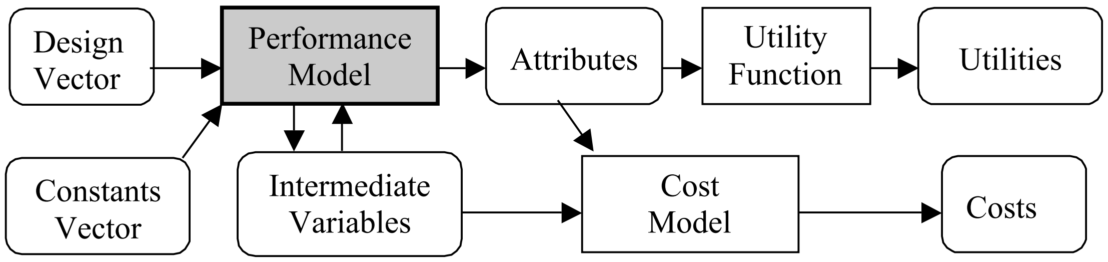


Figure 8-1 Modeling approach

<u>Organization of data</u>

The *design vector* is an enumerated set of design variables, capturing a large but finite number of possible systems. The *constants vector* is an set of values that are fixed, at least for the moment; they include both true constants (i.e. physical or mathematical values such as pi, material properties, and the like), variables that could have been considered as design variables but for a variety of reasons were fixed at a single value, and values that quantify assumptions made about the tradespace. It is important that all of these types of variables be included in a defined constants vector, rather than hardwired into the code, for a variety of reasons, including the need to run sensitivity studies and the possibility that some constants may need to be upgraded to design variables as the tradespace analysis progresses. The *intermediate variables* include values that are not necessarily attributes, but are interesting in and of themselves, or necessary to calculate costs. Typical intermediate variables include system mass and power, orbit parameters, launch vehicle selections, and the like. As with the constants vector, it is important that these be identified explicitly rather than built into the coding of the model, as they may be called for as additional outputs to help understand the tradespace, be required as inputs to additional models, used as interfaces between models, or be upgraded to attributes. The *attributes* and the associated *utilities* and *costs* are the outputs of the models; they are calculated for every possible system evaluate. This set of attributes, utilities, and costs, make up the *solution space* of the evaluated *tradespace*.

8.2.Creating models

The first step in creating the models is to decide what physics, operations, etc. are to be modeled. This is necessarily going to be experience-based to some degree. The QFD matrices created in the tradespace definition can help. These matrices rated the likely affect, (determined using experience, prior knowledge and common sense) of each of the design variables on each of the attributes. The strong effects must be modeled; the more of the weaker ones that can be modeled practically, the better. The question to ask is why are the links strong (or there at all) and what is the controlling physics or other causes that must be modeled.

The X-TOS example below gives a good example of extracting things that need to be modeled from the QFD. In general, these are not surprising; orbital dynamics, basic parameters (mass, power, communications capabilities) of the vehicles, and mission sequences need to be calculated or simulated. Models of appropriate fidelity must be obtained or written for each, and the overall model integrated.

The main issue facing designers creating a model is the trade-off between accurately determining the utility/cost of a particular architecture by having a high fidelity model versus accurately mapping the contours of the tradespace so as to locate regions of high value. Given fixed computational resources one should generally start with a fast inaccurate model in order to get a good feeling of the contours of the tradespace and then, as concepts are eliminated, improve fidelity while focusing only on previously identified high value regions. Some iteration of both the code and the calculations will be needed since higher fidelity may make visible undesirable traits of architectures, hence effective version control and model management are an absolute necessity. A modular software architecture should be used to allow easy migration to higher fidelity codes—see any good software engineering text for details. The first pass on the model should focus on simplicity of the model not on high fidelity or speed. Then as the tradespace exploration progresses, tools such as a profiler should be used to focus optimization efforts of key bottlenecks. Remember that optimized code is often hard to read/debug. Optimize the code only where it is needed.

Organizing models – the Design Structure Matrix (DSM)[4]

Under most circumstances, the models need to be linked to complete the calculation of all of the attributes. The Design Structure Matrix technique has proven very useful for performing this integration in a logical way. This tool is presented very well on the MIT DSM website, www.DSMweb.org. A tutorial is available there, which is required reading if you wish to understand this tool. Rather than duplicate the tutorial, we will give the briefest possible description of the DSM and show how it can be used in the X-TOS example, below.

The DSM is a square matrix with N rows and columns, one for each of N models to be integrated. Information flow between the models is shown by dots in the cells of the matrix. If model m provides information to model n, a dot is placed in the cell in column m, row n. One hopes that information would flow from earlier models to later ones. If the models are arranged in nominally chronological order, having dots only below the diagonal means information is flowing only from early models to later ones. A dot above the diagonal represents information flowing from a later model to an earlier one—requiring iteration. The further away from the diagonal the dot is, the longer the iteration loop will be. Other aspects of the information flow, such as branching and recombining, models running independently in parallel, or processes that are fully coupled to each other, can all be seen at a glance on this kind of chart.[*]

A note on iteration and optimizations

Ideally, the models created would explicitly simulate the performance of the system, and allow direct calculation of the attributes. A good, although computation-intensive, example is orbital dynamics – given the orbit parameters, the orbits are propagated, and coverage and other attributes of interest are calculated. Sometimes, however, iteration is required. Iterations come in two flavors; coupled non-linear models that require iteration for solution, and iterations to find optimal solutions to parts of the problem.

Some physics is non-linear, and unless the problems being solved are very simple, iterative solutions will be necessary. Given unavoidable iterations, the best modeling practice is to keep the iterative loops small and local. A local iteration to, say, roughly size a thermal radiator, contained in a single code module, will have little effect on the overall model speed. The same iteration carried out by linking linear models of the entire spacecraft system, and the iterating the integrated model set, will result in computational difficulties if one wishes to check many possible designs.

Some iteration may be necessary because existing models are interdependent, e.g. sizing a power system may increase the system mass, requiring a larger bus, and hence more power. As modeling becomes more detailed, this kind of iteration becomes inevitable (see the week on ICE to follow). However, this kind of iteration can be detrimental to the efficiency of MATE models, and should be eliminated where possible. Using rules of thumb, margins, or linearized influence factors can provide acceptable approximations for interdependencies of moderate strength; weak ones may be simply ignored, if it can be shown (through sensitivity studies, see below) that they are unimportant to the final result.

[*] Unfortunately, there is no fixed convention as to which side of the diagonal represents "forward" information flow; some published DSM's must be read the opposite way from our example! Make sure to check for any given authors convention, and to specify yours when using DSM's in publications.

As an example of the above, consider computing the lifetime of a spacecraft. The lifetime depends upon the number of times the batteries are charged and discharged. The number of charge cycles is dependent upon the efficiency of solar arrays. The solar array efficiency is dependent upon the level of degradation of the panels. The level of degradation is dependent upon the lifetime. Thus to compute the lifetime one needs to know the lifetime. The model can iterate over various guesses for the lifetime until it converges. This iteration can be very time consuming and should be avoided early in the design process when evaluating a large portion of the tradespace is paramount. One way to eliminate such an iterative loop is to set one or more of the intermediate variables such that a bound can be established on the desired output. For example one could design the solar arrays to be operational for the maximum useful lifespan. When the lifespan is shorter than this maximum, the solar arrays will have been over designed but one will still have an upper bound on the cost incurred to deliver the computed lifespan. If iterations loop cannot be avoided tools such as a DSM or design structure matrix can be used to rearrange the order in which code modules are called so as to minimize the size of the loops. Remember that DSM can only rearrange existing code modules; one should also look for alternative code structures. For example, one could make lifetime a design variable instead of say fuel mass thus eliminating the loop in computing lifetime.

Careful consideration must be given to the other flavor of iteration – the optimization of parts of the problem. In general, the MATE method does not optimize solutions (see Week 8 material for exceptions to this rule); instead, many solutions are ranked (through utilities) and explored. However, it may be useful to optimize for certain intermediate parts of the problem. These are typically intermediate variables such as vehicle bus parameters, orbit parameters, and the like, which affect the attributes but are not in the design or constants vectors. The desire is then to select, for each system considered, the "best" values based on some rational criteria. These may be simple (minimize cost) or more complex (optimize for one or more of the attributes). There is not simple rule of thumb for how handle these sorts of optimizations. An approach that is consistent with the MATE approach would be to optimize intermediate variables if it is both computationally straightforward, and the optimization criteria is simple and requires no decision-maker input. The later point implies that the optimization be a "no-brainer"—reducing cost or improving a single attribute without affecting any others are examples. If the proposed optimization is either computationally difficult, or affects multiple attributes, it should be avoided either by recasting the model, or by adding design vector elements so that the effects of the variables in question can be viewed on the trade space.

Tools for creating models

No attempt at a complete survey of modeling tools will be attempted here. However, the experience of the SSAPRC design teams will be related.

Commercial tools exist for the modeling of space systems. The principle tool used by the SSPARC teams was Satellite Tool Kit from Analytical Graphics (see http://www.stk.com/). Several other tool kits for space vehicle sizing were considered, but ultimately not used; it was found that the simple methods put forth in Chapters 10 and later in SMAD[5] were an excellent resource, and could be coded as required fairly easily.

A variety of tools and frameworks are available to aide in code development. Most of the implementations of the MATE process to date have used MATLAB (http://www.mathworks.com/). Its internal ability to handle large vectors matrices is quite useful when exploring tradespace consisting of thousands of design. Oculus's CO-DOME (http://www.oculustech.com/co/) protocol can be used to split computation among several machines. Phoenix Integration's ModelCenter package has proved useful for organizing and reusing computational modules. Packages are also available to for certain computational tasks. STK (http://www.stk.com/) is a very powerful orbital dynamics calculator it can be particular useful for computing coverage related statistics for constellation – a task that can be very tedious to code by hand. DATA from @RISK automates decision trees and can easily be interfaced with Excel (and with MATLAB, though not so easily). The Mathworks Statistics toolbox for MATLAB is very useful for post processing tradespaces. It has several tools for finding non-obvious patterns. Hierarchical clustering can be very useful in finding groupings. See the toolbox documentation for more information.

Though the software packages mentioned above are very useful, some care should be exercised when using them. Don't treat them as black boxes. Familiarity with the assumptions made and algorithms used in the software packages can be crucial for tracing errors and interpreting results. X-TOS offers an instructive example of this. Originally, orbit computations were done in STK. Small but significant errors were soon observed in the output – orbits that should have been circles were becoming slightly elliptic. After much investigation, these errors were traced to a difference in certain earth constants between STK and the MATLAB interface to STK. To help diagnose the problem, an orbit computer than ran entirely in MATLAB was developed. Though not as sophisticated as the STK orbit computer, it did provide the needed information. The X-TOS team soon realized that most of the information provided by STK was not relevant to the X-TOS problem and replaced STK with the MATLAB only solution. The lesson here is to be very familiar with the tools being used.

The process of actually writing quality code is much like good writing in that it is as much an art as a science. Two useful references are *Code Complete* by Steve McConell and Mastering MATLAB by Hanselmen and Littlefield. McConnell provides a methodology for turning algorithms into bug-free code that runs successfully the *first* and every time that it is run. Useful topics include interface management, error trapping and recovery and optimization. Hanselmen is particularly useful when developing in MATLAB. It provides a clear explanation of the vector processing capabilities in MATLAB and suggestions on exploiting them. Vector processing can often speed up code by an order of magnitude or more. In addition there are many software engineering methodologies available. To date, their usefulness in MATE is a largely topic for future research.

Once the code is completed, some model verification should be done. Given the non-linear and possibly non-deterministic nature of these models, it can be difficult if not impossible to prove their correctness in the general case. When such proof is impossible, a cased based approach should be used. First a series of 'typical' designs should be tested. Then, extreme cases should be tried. This especially important if the model includes any optimization. Extreme value testing may reveal missing constraints. An example was a 60 foot long vehicle, dubbed the 'Telephone pole', during testing of the X-TOS code, created due to missing size constraint. Testing should first be done at the module level. The number one rule for code testing is "**The suite of test cases should *at a minimum* run every line of code written.**" Once module level verification and testing is completed, the module level code can be frozen and system level testing commences. At the system level, error handling becomes very important. The computer not having years of S/C design experience is likely to produce designs that are impossible to implement. Evaluating tradespace often takes a very long times hence the simulation should not be brought to halt by an "impossible design." Rather, it should record an error and then move to other points in the tradespace. Recording the error is key since the error data can be post processed to identify the feasible region of the tradespace. Sensitivity analysis should also be done so that errors can be estimated.

A note on multi-concept modeling

If the tradespace includes more than one distinct concept, more than one set of models may be necessary to fully populate the tradespace. A simple example would be a tradespace including both space-based and airborne options. Clearly different models would be need to calculate coverage in the space case (dominate by orbital mechanics) and the airborne case (dominated by basing, range, and speed of the craft). This complicates the modeling, but does not fundamentally change the approach.

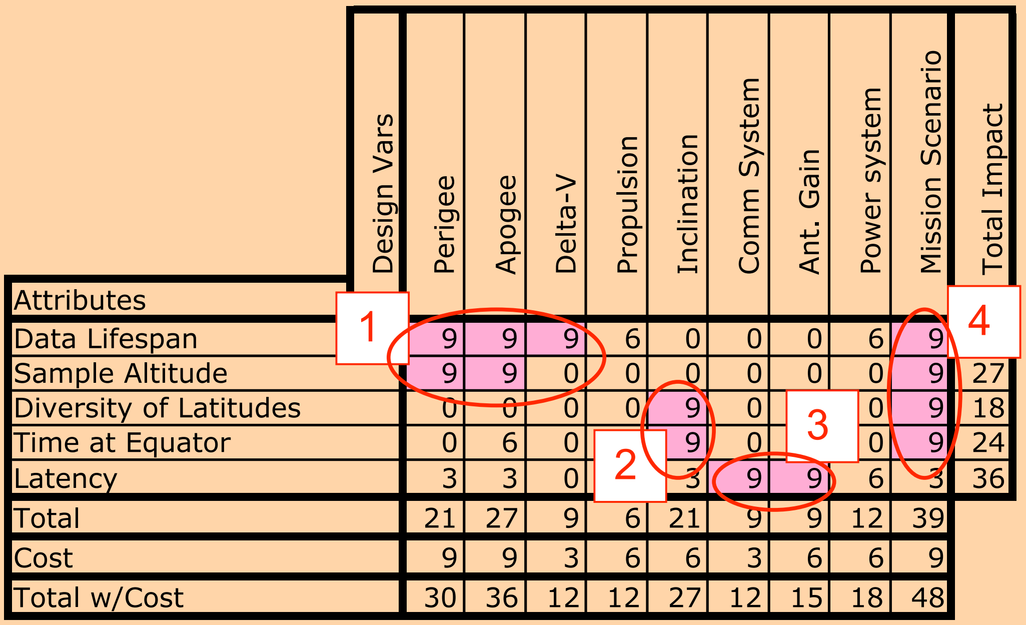
8.3.X-TOS Example

Figure 8-2 shows the X-TOS QFD. The blocks of strong interactions include:

1) The complex orbital dynamics involved in maintaining a low orbit, dominated by drag, and requiring both knowledge of the orbital parameters and some details about the vehicle, e.g. its mass, shape, and delta-V capability.
2) The simpler orbital dynamics relating the latitudes passed over to the orbit inclination
3) The design of the communications system, both on the satellite and in the relay, ground reception, and ground data handling
4) A lifetime simulation of the missions, tying the number of vehicles used an their various orbits, launch times, etc. to the data collected.

Secondary effects (6's and 3's on the DSM) include the effect of the spacecraft power and propulsion design on mission lifetime, the effects of available power on communication capability, and the effect of orbital elements on data latency (through the communication range and interaction with relay points).

In designing the X-TOS model, the team was faced with far more possible missions than could be practically modeled in the short time available. Attention was therefore focused on reducing the scope of tradespace exploration to a manageable size. A series of 'expert rules' to select missions that are likely to yield high utility were created. Embedding such judgment into the model runs counter to the philosophy of the MATECON process. The purpose of tradespace exploration is to explore regions of the design space that were missed by traditional design methods. However, as will be seen when the tradespace is explored in the next week, no harm was done in this case.



Attributes / Design Vars	Perigee	Apogee	Delta-V	Propulsion	Inclination	Comm System	Ant. Gain	Power system	Mission Scenario	Total Impact
Data Lifespan	9	9	9	6	0	0	0	6	9	
Sample Altitude	9	9	0	0	0	0	0	0	9	27
Diversity of Latitudes	0	0	0	0	9	0		0	9	18
Time at Equator	0	6	0		9	0		0	9	24
Latency	3	3	0		3	9	9	6	3	36
Total	21	27	9	6	21	9	9	12	39	
Cost	9	9	3	6	6	3	6	6	9	
Total w/Cost	30	36	12	12	27	12	15	18	48	

Figure 8-2 Examining the Attribute/Design Vector QFD Matrix

X-TOS applied the following rules:

1) Use physically identical satellites for a given mission – *Don't need to explore all possible parings of satellite designs*
2) No more than two satellites per mission – *Reduced the impact of combinatorial explosion with muli-S/C missions*
3) No more than one satellite per launch vehicle – *S/C sizing does not need to account for other S/C in mission*
4) Several mission specific rules that are not relevant here

In addition to the advantages listed with each rules, it is important to understand the constraint each implies on the tradespace exploration. For example, the first eliminates the possibility of having a two S/C mission with one S/C optimized for high altitude long duration flight (i.e. larger; with more stores) and another optimized for low altitude flight (i.e. smaller and more compact). Such a scenario seems quite attractive in terms of the user attributes. Without knowing a priori the characteristics of good high and low altitude design, the computational costs prevented consideration of this case during MATE. It can however be explored during CON with 'good' high and low altitude design are available.

Even though rule 1 eliminated the need to explore combing physically different S/C, different orbit combination still remained a challenge. If a S/C needed to be recomputed each time the orbit or *another* S/C in the same mission changed, computational costs would again explode. To gain better insight into this and other problems, the team first built a very basic single S/C model. Out of this came two key insights: (1) All attributes can be computed for a single S/C orbit combination and (2) For multi-S/C missions, the mission-level attribute values can be determined directly from the individual S/C attributes. These two observation imply that the computation of the single S/C attributes can be decoupled from the mission level attributes. The resulting software architecture is depicted in the block diagram below:

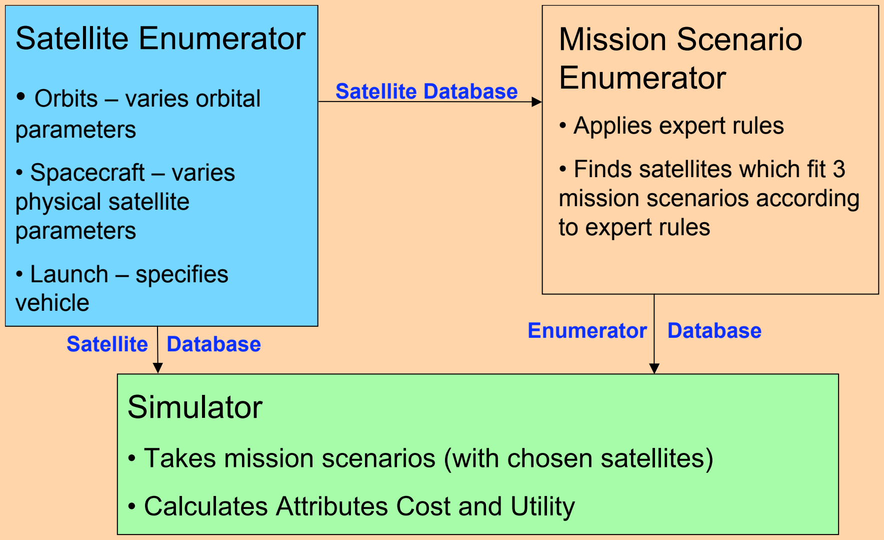

Satellite Enumerator
• Orbits – varies orbital parameters
• Spacecraft – varies physical satellite parameters
• Launch – specifies vehicle
Satellite Database
Mission Scenario Enumerator
• Applies expert rules
• Finds satellites which fit 3 mission scenarios according to expert rules
Satellite Database
Enumerator Database
Simulator
• Takes mission scenarios (with chosen satellites)
• Calculates Attributes Cost and Utility

Figure 8-3 X-TOS Model Top-Level code modules

The resulting DSM also had a advantageous structure:

	Orbit	Spacecraft	Launch	Cost (TFU)	*SATDB*	Mission Scenarios	Cost (Lifecycle)	Calc Attributes	Utility
Orbit	■								
Spacecraft	X	■							
Launch	X	X	■						
Cost (TFU)		X	X	■					
Satellite Database	X	X	X	X	■				
Mission Scenarios	X	X		X	X	■			
Cost (Lifecycle)		X	X	X	X	X	■		
Calc Attributes	X	X			X	X		■	

Figure 8-4 X-TOS Top-Level DSM – Colors correspond to block diagram

First note that all entries are below the diagonal. This means that there are no iteration loops among the modules. Second, note the grayed entries in the lower left. These are additional coupling that were eliminated by separating the computation of mission level attributes from S/C level attributes. By eliminating these entries, the code could be run in three blocks. First, the blue Satellite Enumerator generates the S/C level attributes, then, the red Mission Scenario Enumerator applies expert rules producing combinations of S/C that are likely to yield high utility missions, finally the green Mission Scenario Simulator takes the single S/C attribute values stored in the by the Satellite Enumerator in the SATDB and the scenarios specified by the Mission Scenario Enumerator, and computes total mission attributes (plus utilities and costs) for each enumerated scenario.

Given that each block draws from the previous block, why separate them? The reason stems from the fact that Satellite enumeration is the most expensive portion of the process. Once completed though a variety of different mission level enumeration schemes can be applied without re-computing SATDB. This allows the designers to explore changes at the mission level decisions (e.g. will there be one S/C or two) at little cost. Also the expert rules are contained in the Mission Scenario Enumerator and hence can be modified at little cost as well.

8.4.SPACETUG Example

The spacetug model was simple enough it will be fully laid out here. Although ultimately not much of a modeling challenge, it illustrates the MATE model organization rather well.

The QFD matrix, (see Week 3 material) can be examined quickly. The strong linkages are response time, linked by simple physics to the thrust level of the propulsion system; equipment capability, assumed to be proportional to equipment mass; and Delta-V, a function of all of the design variables. The weak dependence of the response time on fuel and equipment mass was simply ignored. The modeling required was therefore to calculate the available delta-V, and to estimate costs.

The vectors were laid out as follows:

Design Vector		Constants Vector		Intermediate Variables		Attributes	
Pt	Propulsion system (4 types)	c_d	Dry mass cost coefficient ($/kg)	M_b	Bus mass (kg)	Δv	Change in velocity (m/sec)
M_c	Mass of observation/ manipulator system (kg)	c_w	Wet mass cost coefficient ($/kg)	M_d	Dry mass (kg)	RT	Response time
M_f	Fuel mass (kg)	I_{sp}	Specific impulse (sec) (for each *Pt)*	M_p	Mass of propulsion system (kg)	EC	Equipment capability
		g	Acceleration due to gravity (9.8 m/sec^2)	M_w	Wet mass (kg)		
		m_{bf}	Bus mass fraction coefficient				
		m_{p0}	Propulsion system base mass (kg) (for each *Pt)t*				
		m_{pf}	Propulsion system mass fraction coefficient (for each *Pt)*				

The constants vector consisted of simple design rule-of-thumb coefficients for the mass of the propulsion and bus systems on the vehicle, assumed I_{sp}'s for the 4 types of propulsion systems considered, and simple costing coefficients. These were fixed, but could be changed at will to study the sensitivity of the model to them.

The intermediate variables calculated were the major vehicle masses. These were used to both calculate the Delta-V (along with the fuel mass from the design vector and the I_{sp} and g from the constants vector) and to calculate the cost using a very simple parametric model. All calculations were explicit, i.e. the intermediate variables were calculated directly from the design vector and constants vector, and the attributes were calculated directly from the intermediate variables and the two vectors. The details of the model are found in Reference[6]. The model was simple enough to be implemented in an Excel spreadsheet.

Given the uncertainties inherent in the highly conceptual SpaceTug vehicles, a simple parametric model was deemed most appropriate. The equations used in the SpaceTug model follow. This very simple model was found to be remarkably accurate when backed up by the much more detailed ICE analysis—see the the SpaceTug ICE analysis to follow next week.

The response time attribute is solely dependent on the propulsion system selected

$$RT = 0 \text{ if } Pt = \text{electric, otherwise } 1 \tag{8-1}$$

The equipment capability is taken to be proportional to the equipment mass

$$EC = M_c \tag{8-2}$$

The total mass of the propulsion system is taken to be

$$M_p = m_{p0} + m_{pf} M_f \tag{8-3}$$

The vehicle bus mass is calculated as

$$M_b = M_p + m_{bf} M_c \tag{8-4}$$

The vehicle dry mass is calculated as

$$M_d = M_b + M_c \tag{8-5}$$

and the vehicle wet mass is

$$M_w = M_d + M_f \tag{8-6}$$

The total delta-V attribute is then

$$\Delta v = g\, I_{sp} \ln\left(M_w / M_d\right) \tag{8-7}$$

Note that the above delta-V is the delta-V the vehicle imparts on *itself*; the relation of this attribute to useful delta-V applied to target vehicles is complex and mission dependent. The above measure is a useful mission-independent metric of delta-V capability. The first-unit delivered cost is estimated based on a simple rule-of-thumb formula.

$$C = c_w\, M_w + c_d M_d \tag{8-8}$$

9. EXPLORING THE TRADESPACE

9.1.Goal of Tradespace Exploration

The goal of tradespace exploration is to understand the space of possible solutions in depth. This knowledge is useful not just for finding the "best" solution – it is to be used to make the best decisions about a range of topics: the right solution(s) to pursue, the risks to mitigate, the trades to negotiate, and new possibilities to explore. A partial list of what the tradespace should reveal:

- The feasible solution set
- The best solutions, without preconceptions (open to surprises)
- The underlying physics and its impact on the tradespace
- The underlying design trades
- Robust solutions (not overly sensitive to assumptions, needs, or modeling)
- Insight into other issues (uncertainty, risk, evolution/upgradability, etc)
- Avenues for further exploration, either by MATE or more detailed studies

Understanding the feasible solution set is a simple but important use of a tradespace exploration. Assuming the models used to calculate the attributes are reasonably accurate, and the design vector reasonably open, the attributes calculated will represent a survey of what is possible. In some cases, this may not satisfy user needs, pointing out a problem with the solutions considered, the realism of the users expectations, or the need for further development of technology or product concepts.

The feasible solution set may conversely contain solutions that meet or exceed the users' needs. Finding the best or most cost-effective solutions is of course desirable, if only to provide a good benchmark for further study. Finding the "best" solution is particularly rewarding if it is *not* the design that the users or designers had in mind at the onset of the analysis. Typically, the tradespace will reveal the *Pareto front*—the set of solutions which provide the best utility over a range of costs.

The underlying physics are not usually revealed directly by a tradespace exploration. Understanding the physics usually requires querying of the physical models behind the tradespace. Tradespace features may, however, prompt the asking of the right questions. Gross features of the tradespace (e.g. regions without feasible solutions, "walls" where attributes or costs become very sensitive, etc.) usually have simple physical causes. Important details of the tradespace (e.g. the shape of the Pareto front) may have simple causes or complex ones. In either case, asking the focused question "what causes this feature of the tradespace?" is more likely to lead to important physical insights than the more general "how does this all work?"

The key design trades are perhaps the single most important output of the tradespace analysis. Determining them is something of an art, but the tradespace is a quantitative aid to the art. For example, trades that only effect dominated solutions are not very important. Conversely, the trade or set of linked trades that causes the solutions to move along the Pareto front are the ones that must be presented clearly to decision makers.

The tradespace may also provide insight into a number of important characteristics of proposed solutions. For example, the tradespace may reveal solutions that are particularly robust; i.e. that can deliver good value despite shifts in externalities beyond the control of the developers of the system. Solutions that can readily be upgraded (for example, by changing one design variable) can also be found. The effects of technological, policy, funding, schedule or market uncertainties can also be studied using the tradespace. These sorts of analysis are, to date, more art than science. The tradespace exploration may help to advance these arts.

Note that the inclusion of speculative technology in the models can "stretch" the feasible solution set. This is a valid way of exploring technological solutions to the user's problems. Obviously, it has the danger of making solutions based on technology that may or not be available when needed *look* feasible.

Finally, the tradespace may function as a communication tool between the analyst, designer, users and other stakeholders. The understanding of the problem revealed in the tradespace studies may well change stakeholders impressions of what is possible, feasible, or necessary, and hence change the basic premises on which the original tradespace analyses are based. This is a good thing – it allows unrealistic systems to be rejected, good ones to be improved, and key problems to be mitigated very early in a potential program. The tradespace studies may need to be iterated in these cases.

All of the above explores some of the reasons *why* one might want to do a tradespace exploration. The issue of *how* the exploration should be done is as yet not fully answered. From here, we will proceed by using examples to illustrate various ways of interrogating the tradespace and extracting knowledge from the data it contains. A brief discussion of generic methods will be followed by an extensive section of examples. Use this section to be inspired to explore the tradespace, but do not be constrained to the only the ideas or methods presented here.

9.2.Techniques: Be Curious!

The evaluated tradespace contains a vast amount of data. It contains hundreds or thousands of designs, each specified by a design vector of perhaps ten elements, each characterized by perhaps five attributes, and each analyzed with the help of perhaps dozens of intermediate variables. Turning this data into knowledge is the goal of tradespace exploration.

We will assume that the analyst has the appropriate tools available, and not discuss them further. General purpose analysis programs such as Excel™ and Matlab™ have sufficient ability to display data for the purposes of this discussion; most of the examples were generated with these tools. More advanced data visualization tools may be of interest; numerous advanced packages are available commercially, or software specialized to the display of tradespaces may be used. Stump *et al.*[7] documents one such system; another is used in the next section to explore our example systems.

<u>The Big Picture</u>

The first step is to get a "big picture" of the tradespace. The obvious place to start is the two ultimate products of the MATE process, the utility and the cost. The classic "tradespace" plot consists of a scatter plot, with a point for each evaluated design, on axes of cost and multi-attribute utility. There is no convention as to which goes on what axis – both are dependent variables. The gross features of the tradespace should be visible on this plot. Where the Pareto front is, what shape it has, and whether reasonable utilities are achieved should be visible. Features such as large blocks of dominated designs or "islands" of designs separated from the rest should also stand out.

The big picture plot can be used to explore somewhat deeper into the tradespace by the use of various texturing techniques. Color, size, and shape of the icons representing each design can be used to isolate various aspects of the tradespace and view their effects on the utility and cost. For example, the value of a single design variable could be represented by various colors, creating a plot that highlights the role of that design variable. The additional features plotted using the extra "dimensions" could include:

- Values of a given attribute. This shows effect of that attribute on multi-attribute utility and cost. It should highlight critical attributes, and point out ones that prove non-discriminating.
- Values of a given design variable. This shows effects of design choices on the multi-attribute utility and cost. It should highlight the critical variables, point out the ones that prove non-discriminating, and provide the user with insight into the trades and/or design choices he or she must pursue to increase utility.
- Values of an intermediate variable. This can be used to gain a variety of insights into the physics and/or utility trades that control the tradespace.

Projected three-dimensional plots can also be used, although experience suggests these will be relatively difficult to use. The Spacetug and X-TOS examples illustrate many of the above techniques for displaying the tradespace.

The big plot can also be explored in more detail by cropping (looking at only a restricted range of values of costs and utility), stretching, or rescaling (for example, to a log scale). We are most interested in the ranking of the potential solutions, and the cost/utility tradeoffs involved, so altering scales can be done in any way that highlights this information. Good examples of the use of scaling are given in the X-TOS and B-TOS example sections.

Plots of single attribute utilities or their associated attributes against cost, other utilities, or design variables also may be useful. The A-TOS example shows the two meaningful single-attribute utilities plotted against each other to clarify the implicit trade between the two. This aspect of tradespace exploration is under-represented in our examples; do not hesitate to try combinations that might provide extra value.

The Pareto Front

The feature of most interest on the big-picture plot is the Pareto front. The Pareto front is the set of designs that produce the most utility for a given cost, or conversely have the lowest cost for a given utility. These are the "rational" choices; any design not on the Pareto front may be considered a poor choice, as one can find a design with the same performance (but cheaper) or the same cost (but higher performing) on the front. On the other hand, making decisions between designs on the Pareto front is making a true cost/benefit trade.

The Pareto front should be explored by cropping and magnifying, to understand its structure (see the TPS example), by exploring the physics of the trades that define its shape (see Spacetug and B-TOS), and by investigation of the designs that make up the set (see the TPS example). Exploring the physics defining the shape of the Pareto front often reveals the physical limits of the proposed system, and the true physical tradeoffs that must be made.

The Pareto front is the most important part of the tradespace, but care should be taken not to over-emphasize the exploration of the Pareto front to the detriment of exploring other aspects of the tradespace. First of all, given the uncertainty and inaccuracy inherent in early studies, the designs on the Pareto front may change based on changes in user needs, analysis accuracy, or other factors. Secondly, the need for flexibility, evolvablity, or other "ilities" not directly expressed in the attributes may make the selection of designs not directly on the Pareto front desirable. In both these cases, attention should be broadened to designs "near" the Pareto front, even if they are not right on it. Finally, it must be remembered that a goal of tradespace analysis is the understanding of the entire tradespace, and this understanding may be valuable even far from the Pareto front. An example of the latter is the need to understand why a proposed, and perhaps favored, point design is NOT near the Pareto front.

Parametric and Sensitivity Studies

One of the most powerful tools for understanding the tradespace is to parametrically alter the inputs, and see the resulting changes. Typically, the effects on both the big picture of the tradespace and the Pareto front architectures are examined. This technique can be used for:

- Helping a user that is unsure of the true utility of a system concept. Historically, it is not unusual for potential users to have only a vague understanding of what a system might be good for. A parametric study, varying the user utility over broad ranges, can find matches between user needs and technological solutions. The SpaceTug example to follow provides an illustration of this sort of study.
- Understanding the effects of changes in user needs. By altering the utility functions and their weightings, the stability of the tradespace, and the robustness of the designs in it, to changing user needs can be checked. This can be done to find architectures that are good choices under a variety of possible changes in need, or in direct response to user requests (X-TOS example).
- Understanding the sensitivities of the models to assumptions. Easiest to analyze are those assumptions that lead to specific values being placed in the constants vector. The values

can be changed and the analyses repeated to see the effect on the trade space. With more difficulty, assumptions embedded in the analytical code can also be checked by altering the code.

- Understanding the effects of proposed advanced technologies, cost savings techniques, etc. These can be modeled, and the simulations re-run with them incorporated. Conversely, the effects of technologies failing to perform as expected can be quantified by including degraded performance in the models. Advanced propulsion technologies are included in the Spacetug example with this intent; conversely, the Techsat study surfaced some technology risk issues.
- Understanding the potential for spiral or evolutionary development. Speculative studies such as those suggested for advanced technologies can also check the suitability of architectural choices to be upgraded at a future date with larger budgets, advanced technologies, etc.
- Understanding the effects of policy decisions, funding instability, and market and technical uncertainties of various kinds. These issues are explored at the end of this section.

Note that the traditional use of parametric studies, to explore design trades, is superceded by exploring the tradespace. If the design vector was chosen correctly, all of the basic trades will be present in the tradespace, and will not require specific parametric studies. However, the assumptions behind the setting up of the design space (e.g. the limits of the design vectors and their enumeration, see the week 3 material) may be appropriate subjects for study of this sort.

Looking at Designs in Depth

Looking at the details of the designs will almost certainly be necessary at some point in the tradespace exploration. The idea is to look at a specific point on the tradespace, and understand the design that defines that point, to the full level of detail available in the simulation model. The A-TOS example uses this method to explore what the best and worst designs look like; the Spacetug, B-TOS and TPS examples use it to understand the systems on the Pareto front.

If more detail is required, systems on the Pareto front (or elsewhere in the design space) can be more fully defined using the ICE method. This will be the topic of the next week.

9.3. X-TOS Tradespace[8]

Figure 9-1 shows the "Big Picture" of the X-TOS tradespace. We see two distinct groupings of architectures. Color has been used to display the effects of a design variable, the number of satellites launched. The black architectures represent a single satellite, the red represent two satellites launched in sequence, and the blue represent two satellites launched in parallel. A glance at this chart shows the poor payoff for more than one satellite. This finding was discussed an confirmed with the user. Further discussion will exclude the multi-vehicle designs.

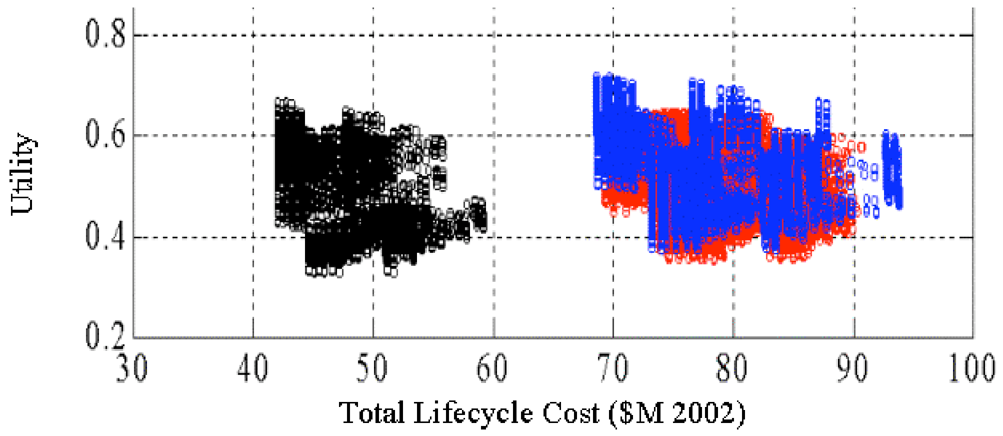


Figure 9-1: Complete tradespace

Figure 9-2 shows the tradespace cropped and rescaled to focus on the single satellite designs. Several distinct horizontal bands can be seen. One of these bands begins at approximately ($51M, 0.33) in the space and continues up and to the right on the graph. This pattern of increasing utility with increased cost is repeated throughout the plot until the top.

At the top, we get a different case. There is a beginning of the regular pattern, but it is truncated before it can rise in cost and utility. The tradespace was restricted at that point due to two factors: the customer's requirement that this mission be launched on a small to medium U.S. launch vehicle and the resulting decision, to remain within the constraints of these launch vehicles, that the design vector only include up to 1,000 m/s of Delta V. Relaxation of these constraints might allow higher cost, higher utility architectures.

Due these constraints we have an unusual Pareto frontier. In this particular case, there is a clear winner: there is a single lowest-cost, highest-utility point on the graph.

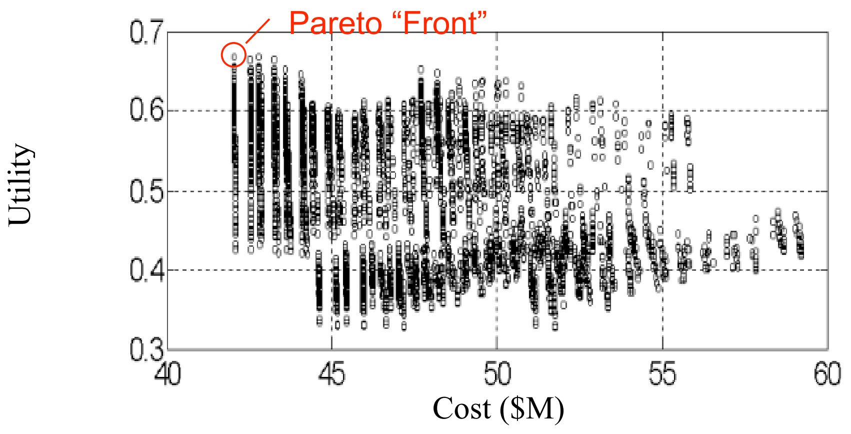


Figure 9-2: Single Satellite architectures

Figure 9-3 and Figure 9-4, when viewed together, tell a story. Each shows the single vehicle portion of the tradespace, with color used to show design vector information. Figure 9-3 is the single satellite utility plot with each architecture colored by its apogee altitude. A strong inverse correlation can be seen, with lower apogee altitudes having a higher utility. Figure 9-4 is a very similar plot, but with the coloring done by perigee altitudes instead of apogee. To clarify, Figure 9-5 shows two enlarged plots of the same architectures (zooming in on the narrow column of points at the far left), with the left plot colored by apogee and the right by perigee. These show that apogee altitude determines what block of the tradespace a design will be in, while perigee determines the order of the architectures *within* these blocks. Within each block is seen the same sort of pattern that we saw within the macro diagram: increasing utility with decreasing perigee.

The absence of any 150 km perigee designs in the tradespace is an important clue. Even with the largest Delta-V in the design vector, no design can meet the minimum requirement for data lifetime at such low altitudes due to the high drag.

Most of the other design variables had only a weak effect on utility, and some effect on cost. These are not shown here. Of these, Delta-V had the strongest effect, so it was selected for further study with the intent of understanding why the tradespace has only as single point on its Pareto “front”.

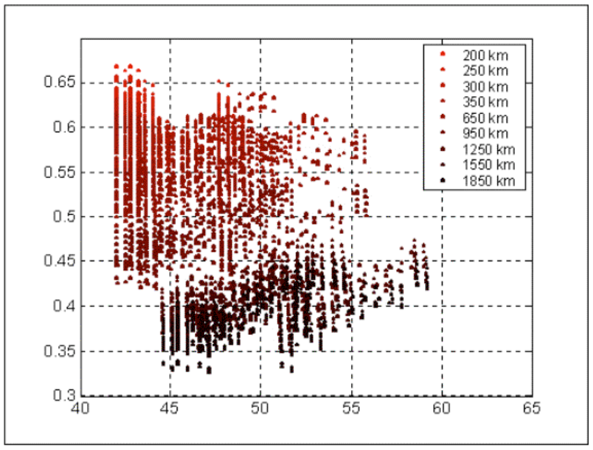


Figure 9-3: Variation due to apogee altitude

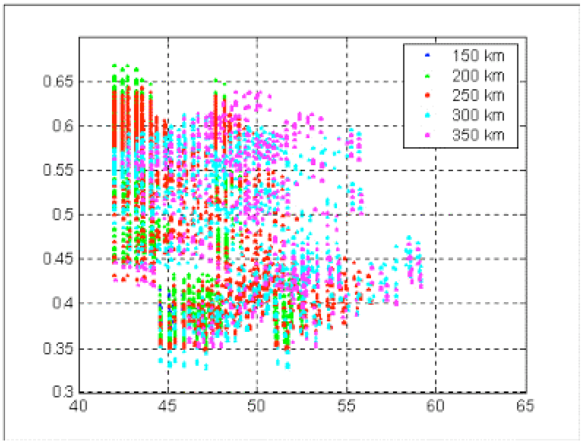
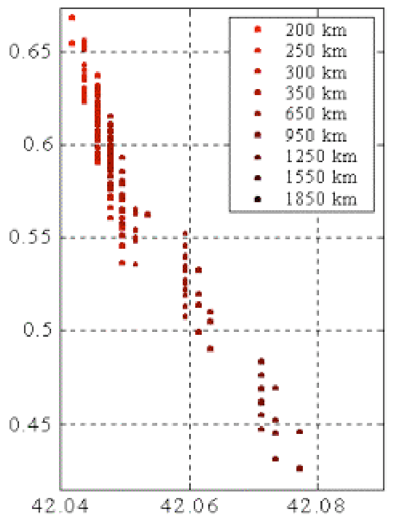


Figure 9-4: Variation due to perigee altitude



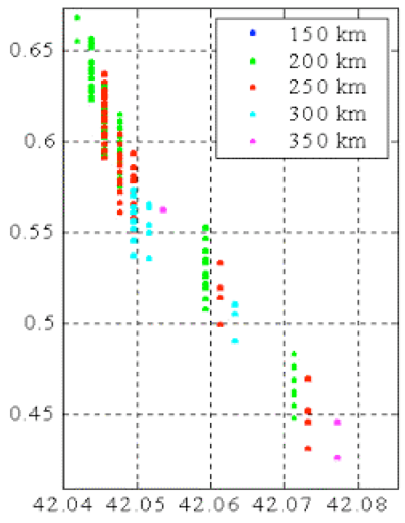


Figure 9-5 Zoom in on left edge

Figure 9-6 plots utility vs. data lifetime (a single attribute) with color used to indicate the amount of Delta V carried (a design vector element). This plot rewards careful consideration. As in the big picture tradespace plots, near horizontal bands are seen (A). The "rainbow" effect shows that these are bands of similar designs, differentiated by delta-V. More delta-V increases lifetime directly. It increases total utility only weakly, due to the low weighting factor given to lifetime in the utility function.

The near-vertical bands of single colors (B) represent isometric lines of delta-V. Within one of these bands, one can reduce the apogee and perigee of the orbits, which moves the architecture up and to the left along the isometric line. Lifetime (and its single attribute utility) is reduced, but total utility is always increased. An expected key trade, between lifetime utility (goes down with altitude) and data altitude utility (goes up as altitude goes down) turns out not to be a trade – it is always better to favor low altitude. This is to be expected, as it comes directly out of the value given by the customer on 'Data Lifetime.' In the original utility function, which was later changed, Data Lifetime was rated as the least important of the attributes.

Hence, to maximize utility it is always desirable to increase fuel load, and decrease altitude. There are limits, however. Line (C) shows the lower limit on acceptable lifetime, six months. Line D shows the upper limit on delta-V for systems in the upper left of this tradespace. These limits create a trap – the best design is hard against them both, with the maximum delta-V (and hence fuel load) and the minimum altitude. It turns out this trap also applies to price; the lowest altitude design is the cheapest, and although the delta-V costs money, no lower delta-V vehicle can meet the minimum life requirement, so no cheaper option is available in the permissible tradespace.

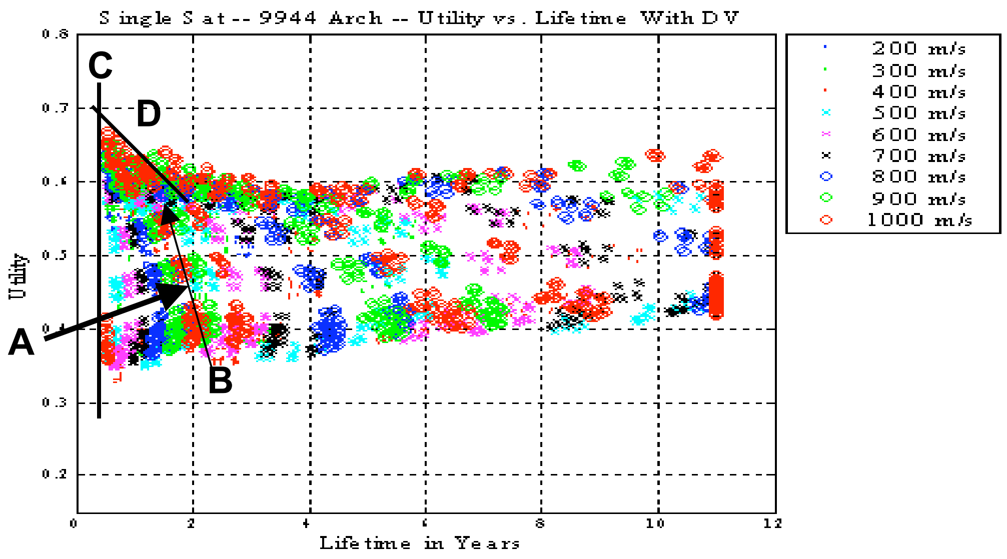


Figure 9-6 Utility vs. lifetime; colored by Delta-V

<u>Re-evaluation of the tradespace with changing user preferences</u>

The X-TOS tradespace model was re-run with new utilities after the above analysis was concluded. In fact, the ICE process (next week) had already begun when the utility team returned to the proxy user to show the selected baseline architecture. Upon seeing the results, the proxy user realized that his preference for lifetime had not been accurately captured. The utility function was reassessed as discussed in Week 4. The new utility weights and single attribute utility curve for data lifespan are shown in a figure in the X-TOS example.

Changing the quantified user preferences at the *end* of the design process would be highly disruptive to traditional design practices. In the MATE-CON environment, they were not. A new utility function was assessed and the architectures were re-evaluated in terms of the new preferences within a day. The difference in the big-picture tradespace plot is shown in Figure 9-7. Under the original utility there was virtually no real Pareto front, and there was no difference in utilities between architectures A, B, and C. Under the revised utility there was a real (and realistic) trade between data lifetime and cost. This trade is expressed by the need to choose among options on the Pareto front seen in the figure. Architecture A is low cost, but has only the minimum lifetime and hence limited utility. Architectures B and C progressively add utility by increasing lifetime; each is progressively more expensive. All are on the Pareto front, and the user must choose between them.

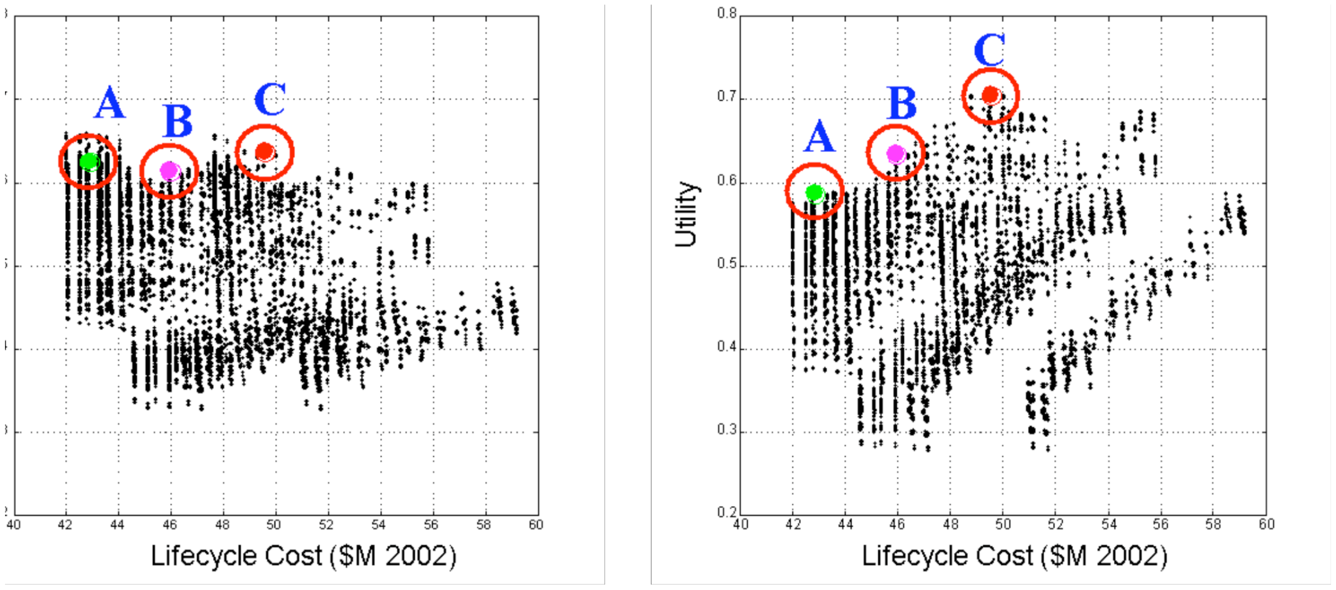


Figure 9-7 Change in the X-TOS tradespace due to changing user utilities.[9]

9.4. Spacetug Tradespace[10]

Figure 9-8 shows the tradespace as a plot of utility vs. cost with each point representing an evaluated design. The Pareto front of desirable designs are down (low cost) and to the right (high performance). The Pareto front features an area of low-cost, lower utility designs (at the bottom of Figure 9-8). In this region, a large number of designs are available, and additional utility can be had with moderate increase in cost. On the other hand, very high levels of utility can only be purchased at great cost (right hand side of plot).

The propulsion system is highlighted in Figure 9-8, with different symbols showing designs with different propulsion systems. The propulsion system is not a discriminator in the low-cost, low utility part of the Pareto front, except that nuclear power is excluded. At the high end, on the other hand, the Pareto front is populated by nuclear-powered designs. Electric propulsion occupies the “knee” region where high utility may be obtained at moderate cost.

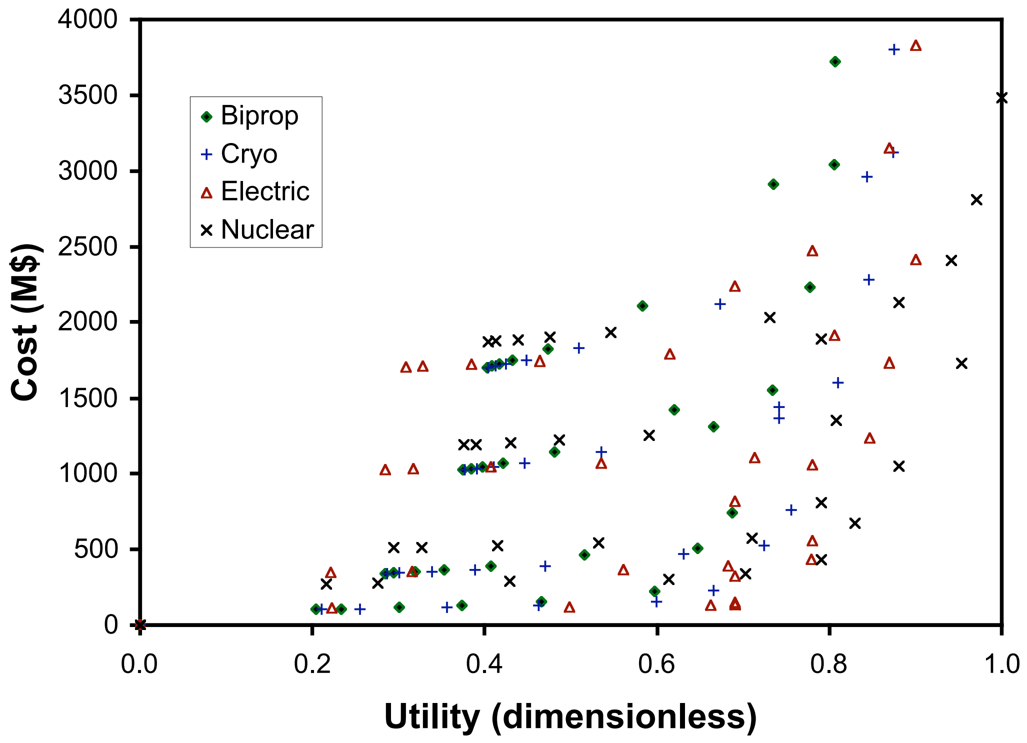


Figure 9-8 Trade space for nominal user, with propulsion system indicated

Figure 9-9 shows the cost banding due to different choices of manipulator mass, or capability. For the lower-performance systems, increased capability translates to large increases in cost with only modest increases in utility. High capabilities are only on the Pareto front for high utility, very high cost systems. This indicates, for the nominal set of user utilities used, cost effective solutions would minimize the mass and power of the observation and manipulation systems carried. Using the utility weights for the "Capability Stressed" user results in Figure 9-10. As expected, increasing capability systems now appear along much of the Pareto front, although capability still comes at a fairly steep price.

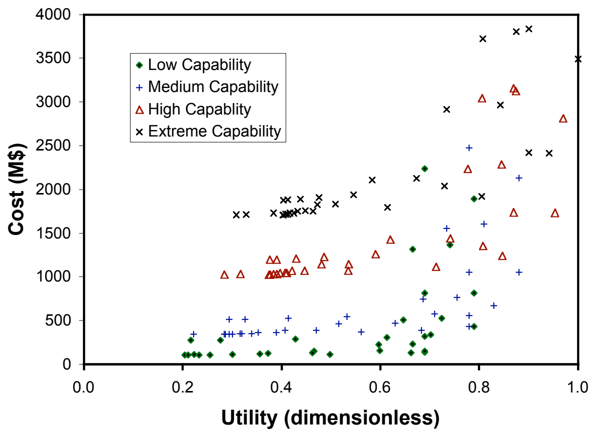


Figure 9-9 Trade space for nominal user, with capability indicated

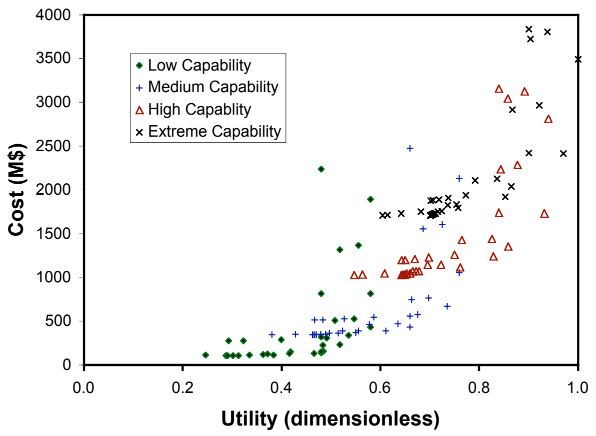


Figure 9-10 Trade Space for capability stressed user

Using the utility weightings for the “Response Time Stressed” user results in Figure 9-11. The results are clear; electric propulsion is eliminated from consideration. In the nominal case (Figure 9-8) electric propulsion appears at the “knee” of the Pareto front, and would appear to give good utility for modest cost, but that conclusion will be very sensitive to the weighting given response time by an actual user. Conversely, if the nominal weights and a delta-V utility function for a user with a demand for very large delta-V the result is Figure 9-12. Now, almost all the designs on the Pareto front feature electric propulsion.

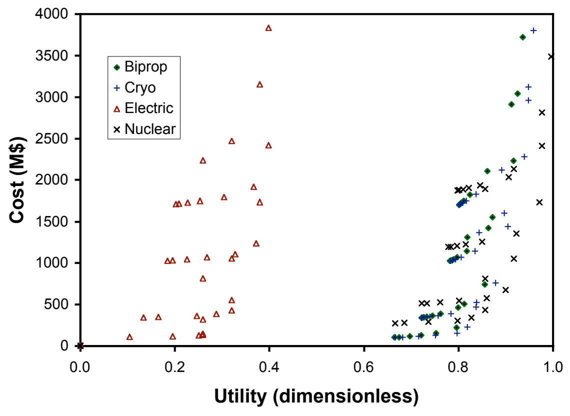


Figure 9-11 Trade space for response time stressed user

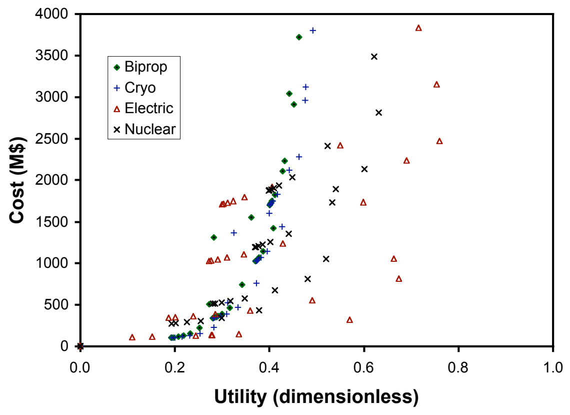


Figure 9-12 Trade space for user with large delta-V needs

A more detailed view of the lower right-hand corner of the nominal Pareto front (from Figure 9-8) is shown in Figure 9-13. Only low-capability systems are shown. The lines connect designs that differ only by fuel load carried. All the propulsion systems appear to hit a "wall" where costs increase sharply at little or no advantage in utility. Examination of the designs on this wall reveal two very different phenomena. The bi-propellant and cryogenically fueled systems are up against the limits of the rocket equation. Each small increment in utility is gained only by carrying a lot more fuel, most of which is used to push fuel around. The nuclear and electric systems, on the other hand, are limited only by the fact that they achieve a high enough delta-V to score a 1.0 on the delta-V utility, and there is simply no value in carrying more fuel. If that limit is removed, both systems show large advantages, as shown in Figure 9-12.

Also shown on Figure 9-13 are some specific designs capable of carrying out the mission mentioned in the introduction—moving from a LEO parking orbit to GEO transfer orbit, grappling a stranded target vehicle, inserting it in GEO, and (optionally) returning to LEO. The biprop design is "on the wall", needing a very large fuel load to create the necessary delta-V. The cryogenically fueled design is not as bad, but is clearly sensitive to the details of its design – slight increases in manipulator mass etc. will send it too "up the wall." Neither chemical fuels can (without refueling) return a vehicle to LEO. The electric vehicles, both one-way "tug" and round-trip "cruiser" do not have this problem. The Electric Cruiser design, in fact, sits in the lower-right corner of the tradespace because it has maximized the delta-V utility, not because it is limited by physics.

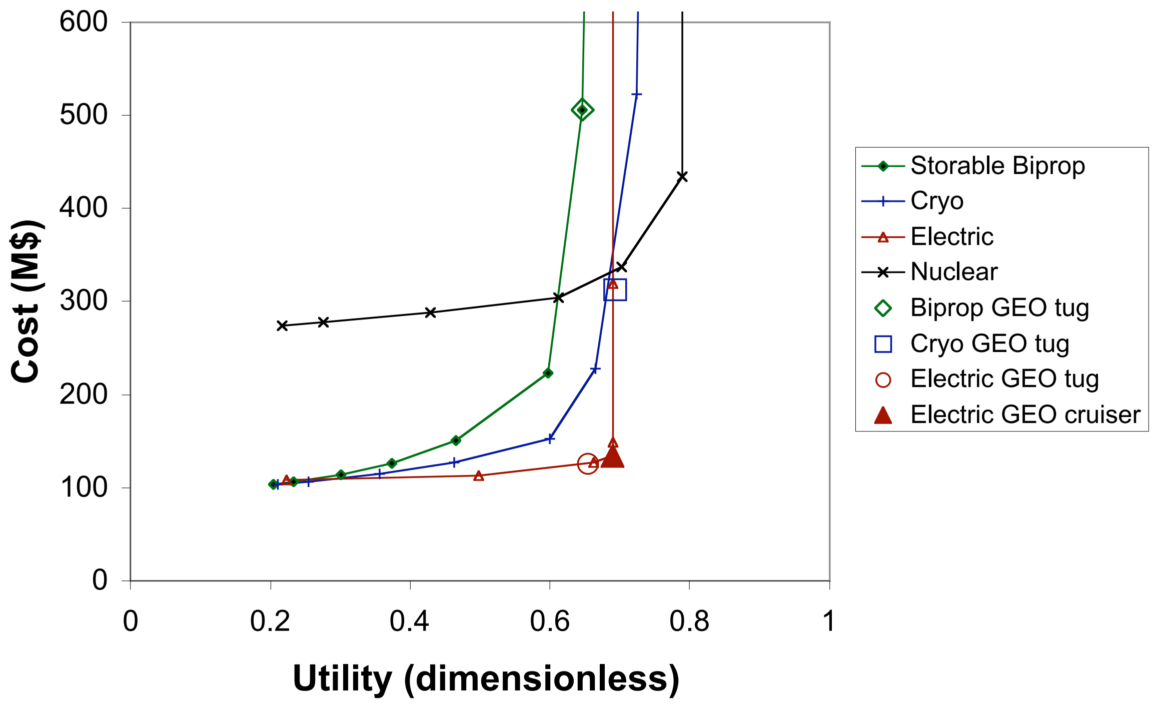


Figure 9-13 Low capability systems, showing effect of increasing fuel load, with GEO rescue vehicles

A final look at the Space Tug tradespace is to examine the actual vehicles represented by the points on the above plots. For this purpose, the intermediate variables, specifically the wet and dry masses of the vehicles, were extracted from the analyses. They are shown in Table 9-1. The effects of the rocket equation "wall" mentioned above may be seen by looking at the wet and dry masses of the Biprop and Cryo one-way vehicles. They are mostly fuel, but the dry weight is also very large (mostly in tankage). The electric vehicles achieve more delta-V with much less mass, although the reaction mass fraction is very large for electric propulsion vehicles, indicating a design challenge. The "tender" vehicles are all lower mass fraction, lower delta-V storable propellant vehicles. Interesting is the LEO 4a tender, which is at the "knee" of the utility-cost curves for biprop vehicles; although it is twice as big as the other tenders, it is still much smaller than the "one-way" vehicles, and achieves almost as much delta-V (and utility).

Note that these designs are based on the very simple model outline in the previous section. The level of detail at which they are examined will be greatly expanded next week.

Table 9-1 Tug Design Summary

Design	Dry Mass (kg)	Wet Mass (kg)	Delta-V (km/s)	Total Utility	Cost (M$)
Biprop one-way	1800	11800	5.5	0.65	510
Cryo one-way	1250	6250	7.1	0.69	310
Electric one-way	710	990	9.8	0.65	130
Electric cruiser	750	1100	12.6	0.69	140
GEO bi-prop tender	740	1900	2.8	0.47	150
LEO 1 tender	690	1400	2.2	0.40	130
LEO 2 tender	670	1200	1.8	0.37	130
LEO 3 tender	650	1000	1.4	0.33	120
LEO 4 tender	720	1700	2.6	0.44	140
LEO 4a tender	980	4100	4.2	0.60	230

9.5.Further Examples of Tradespace Exploration[11]

Terrestrial Planet Finder

The Terrestrial Planet Finder (TPF) study provides an interesting example of exploring the Pareto front. The TPF concept is to use an interferometer to search for terrestrial-type planets in other solar systems. The single performance attribute is number of images collected; the design vector includes viable structural architecture (single or multiple vehicles, creating a one- or two-dimensional array), variable numbers of and sizes of aperture, and variable orbits. When the utility of the candidate systems, reduced to a metric of the number of scientifically useful images taken, was plotted against cost, a clear Pareto front of "best" designs appears on the lower left (Figure 9-14). Along this front, there is a simple trade of performance for money, with the cost/image remaining relatively constant at $0.5M/image. This is good news for the decision maker, who can decide how much to spend on the system (within a range) without worrying that the system will be a poor value. There is a twist not visible on the plot, however. The Pareto front consists of a wide variety of types of systems (see Figure 9-15).

The figure shows the # of images provided by systems on the Pareto front, their cost, their orbit radius, the number of apertures in their design, their structural architecture (connected – SCI, or separated vehicles – SCI, with 1- or 2-D arrays), and the diameter of the apertures. In this case, moving along the front involves changing the architecture of the system, not just tweaking the design. This means that although the decision makers can make a fairly free choice early in the program about what price/performance point they desire, once an architecture is chosen, perturbations in requirements or budgets will tend to result in a non-optimal system.[12]

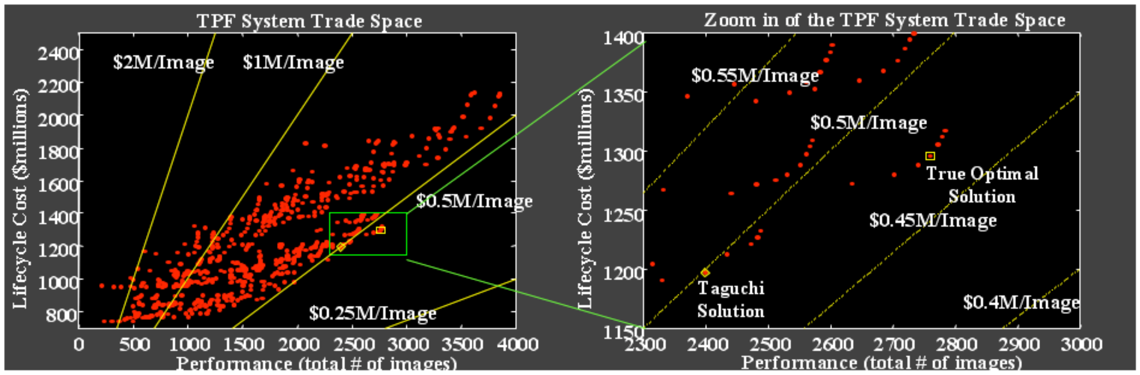
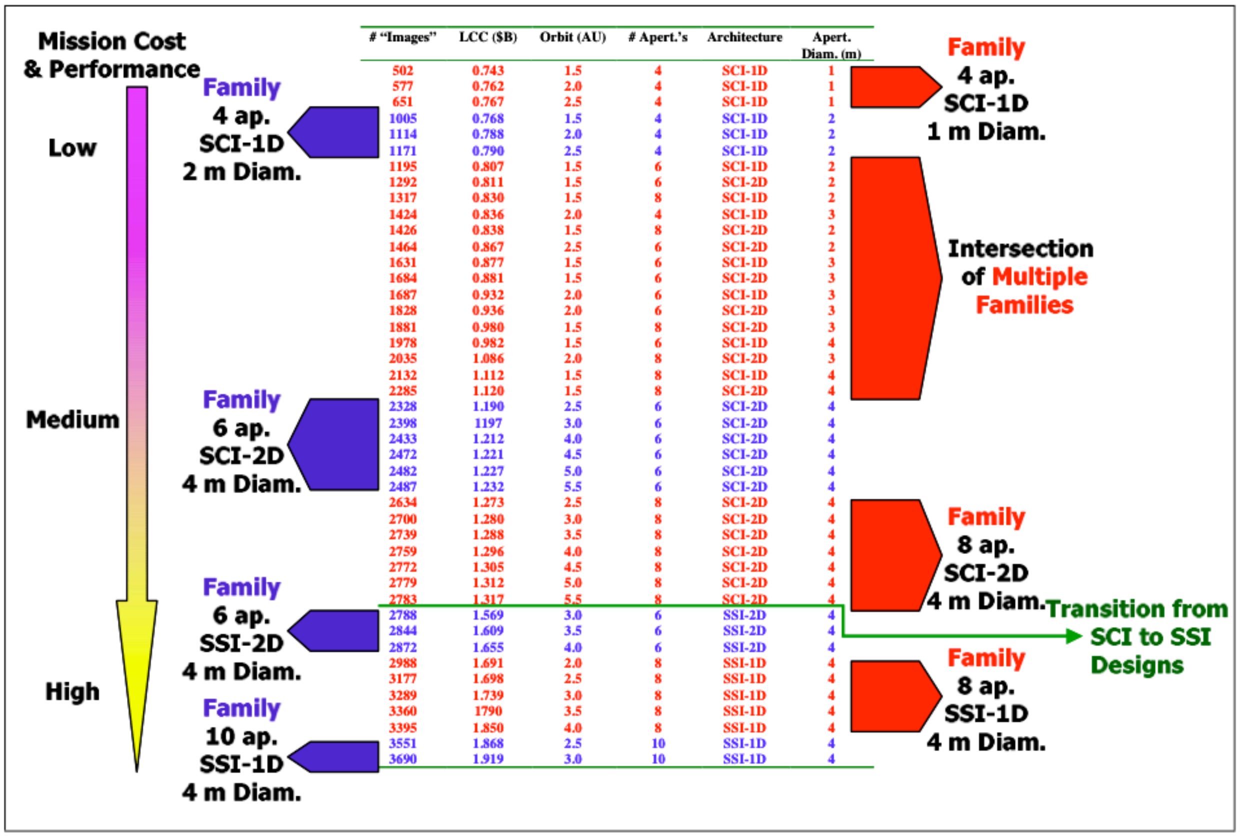


Figure 9-14 TPF tradespace[13]



# "Images"	LCC ($B)	Orbit (AU)	# Apert.'s	Architecture	Apert. Diam. (m)
502	0.743	1.5	4	SCI-1D	1
577	0.762	2.0	4	SCI-1D	1
651	0.767	2.5	4	SCI-1D	1
1005	0.768	1.5	4	SCI-1D	2
1114	0.788	2.0	4	SCI-1D	2
1171	0.790	2.5	4	SCI-1D	2
1195	0.807	1.5	6	SCI-1D	2
1292	0.811	1.5	6	SCI-2D	2
1317	0.830	1.5	8	SCI-1D	2
1424	0.836	2.0	4	SCI-1D	3
1426	0.838	1.5	8	SCI-2D	2
1464	0.867	2.5	6	SCI-2D	2
1631	0.877	1.5	6	SCI-1D	3
1684	0.881	1.5	6	SCI-2D	3
1687	0.932	2.0	6	SCI-1D	3
1828	0.936	2.0	6	SCI-2D	3
1881	0.980	1.5	8	SCI-2D	3
1978	0.982	1.5	6	SCI-1D	4
2035	1.086	2.0	8	SCI-2D	3
2132	1.112	1.5	8	SCI-1D	4
2285	1.120	1.5	8	SCI-2D	4
2328	1.190	2.5	6	SCI-2D	4
2398	1197	3.0	6	SCI-2D	4
2433	1.212	4.0	6	SCI-2D	4
2472	1.221	4.5	6	SCI-2D	4
2482	1.227	5.0	6	SCI-2D	4
2487	1.232	5.5	6	SCI-2D	4
2634	1.273	2.5	8	SCI-2D	4
2700	1.280	3.0	8	SCI-2D	4
2739	1.288	3.5	8	SCI-2D	4
2759	1.296	4.0	8	SCI-2D	4
2772	1.305	4.5	8	SCI-2D	4
2779	1.312	5.0	8	SCI-2D	4
2783	1.317	5.5	8	SCI-2D	4
2788	1.569	3.0	6	SSI-2D	4
2844	1.609	3.5	6	SSI-2D	4
2872	1.655	4.0	6	SSI-2D	4
2988	1.691	2.0	8	SSI-1D	4
3177	1.698	2.5	8	SSI-1D	4
3289	1.739	3.0	8	SSI-1D	4
3360	1790	3.5	8	SSI-1D	4
3395	1.850	4.0	8	SSI-1D	4
3551	1.868	2.5	10	SSI-1D	4
3690	1.919	3.0	10	SSI-1D	4

Figure 9-15 TPF architectures on the Pareto front[14]

Broadband

A generic broadband communication system provides communication services to a generic, distributed market. A study was carried out comparing the utility (in terms of subscriber hours provided) vs. cost of widely disparate spaced-based communication systems. The broadband study provided an excellent example of the need to consider uncertainties when comparing widely disparate systems. The study recapitulated earlier work which showed that MEO or LEO systems could provide much lower cost bandwidth than GEO systems. However, when the uncertainty analysis was included, this low cost was found to be accompanied by huge uncertainties in the results, driven primarily by market uncertainties.[38] This analysis effectively captured the reality of these systems—advanced LEO and MEO systems are currently floundering for lack of market. Figure 9-16 shows a big-picture tradespace, with points surrounded by "uncertainty ovals" due to uncertainty in both performance and cost.

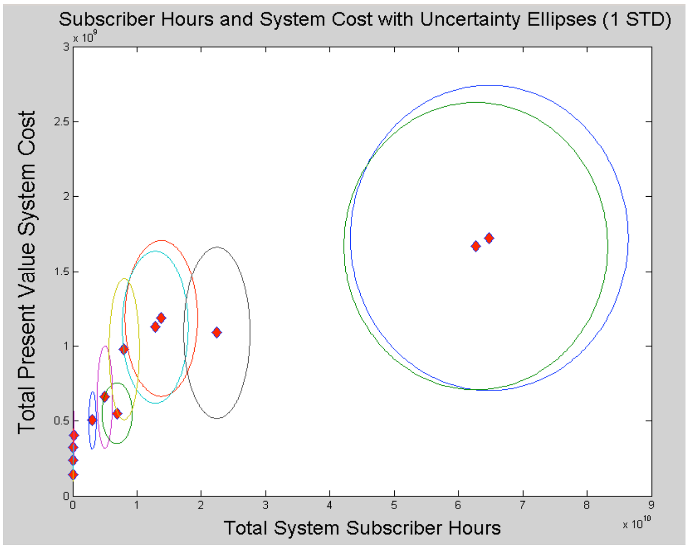


Figure 9-16 Broadband communication system tradespace with uncertainty ovals.

A-TOS

A-TOS (Terrestrial observer swarm A) was a proposed swarm of identical small satellites used to collect ionospheric data. The system was expected to collect three distinct types of data found in different regions of the ionosphere. The A-TOS study was carried out using a prototype version of the current process.[15] It provided a first experience in the *emergent* lessons possible with trade space exploration.[16] Two of the three missions desired for the system proved to be at odds. The "best" systems as determined by the utility function simply ignored the last mission, doing it only nominally while optimizing for the other ones. Useful hybrid missions were possible, but only at fairly severe cost and performance penalties. This information would be very valuable to an up-front decision maker, who could (for example) drop one of the missions at great cost savings early in the program, rather than as a compromise later. Interestingly, it was precisely this tension between the needs of the missions that allowed the effective use of portfolio theory to minimize the risk of the overall program.[17] Figure 9-17 shows the un-normalized utilities for two missions plotted against each other, with shading used to show cost. The "best value" system was number 1, which ignored the High Latitude mission. Insisting on at least some satisfaction of all missions would result in a system such as number 2 being selected; an unfortunate compromise. Note if more resources were available, the best value system could be improved only slightly (to point 3) but a multi-mission system with good value becomes practical (point 4).

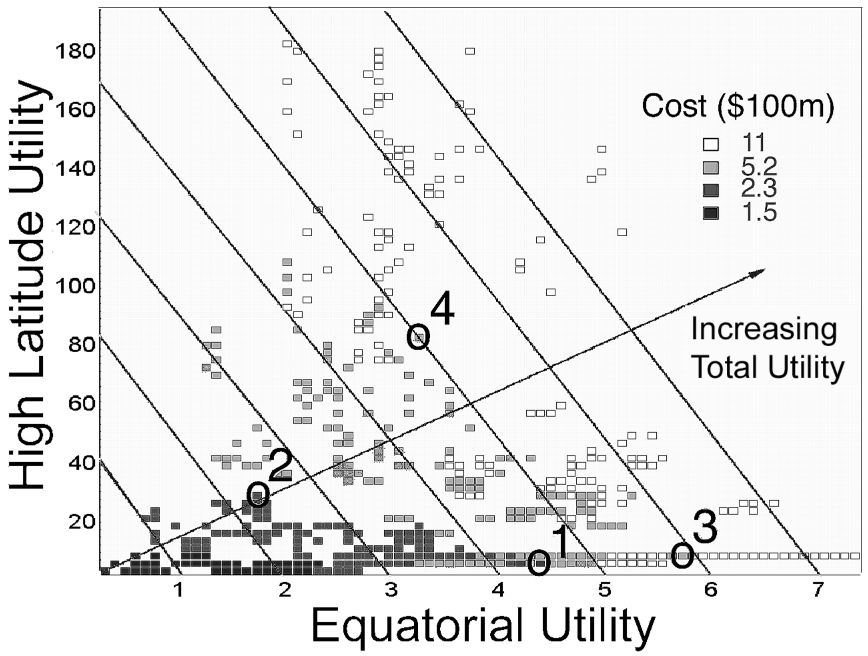


Figure 9-17 A-TOS tradespace showing two mission utilities plotted against each other.

B-TOS

The B-TOS study was the first carried out using formal multi-attribute utility methods. The user needs involved four different measurements. The system architecture included a swarm of small "daughter" vehicles surrounding a central "mother" vehicle. Despite this complexity, the analyses ultimately revealed a simple dominant trade, between the number of daughter vehicles and the accuracy of one of the desired measurements. Other factors proved either impractical (e.g. a desire for global coverage was economically infeasible) or non-discriminating (e.g. other measurements could be performed well by many designs). The result, shown in Figure 9-18, was a Pareto front with only five architectures (out of several thousand) on it. The stair-stepping is due to the physics of the problem—increased performance was only gained by adding complete "rings" of daughter vehicles around the mother ship. This result presents the decision makers with a straightforward set of choices, weeded out of many confounding considerations. It also provides a clear set of possible downstream choices (e.g. dropping vehicles to save money, at a known performance penalty). This clarity made it particularly suitable for a study of mitigating possible policy changes (e.g. funding cuts) using the real options method.[18] B-TOS work was done as part of a graduate space systems class. The MATE method was found to provide a good framework for educating students in system architecture issues.[19]

Note the utility scale, which focuses on values from 0.98 to 1.00, and the cost scale, which is logarithmic. Both were selected to clarify the trades on the pareto front. The unusual utility scale is due the behavior of the multi-attribute utility function–see the Week 4 material under "Understanding Keeney-Raiffa Multi-Attribute Utility Functions."

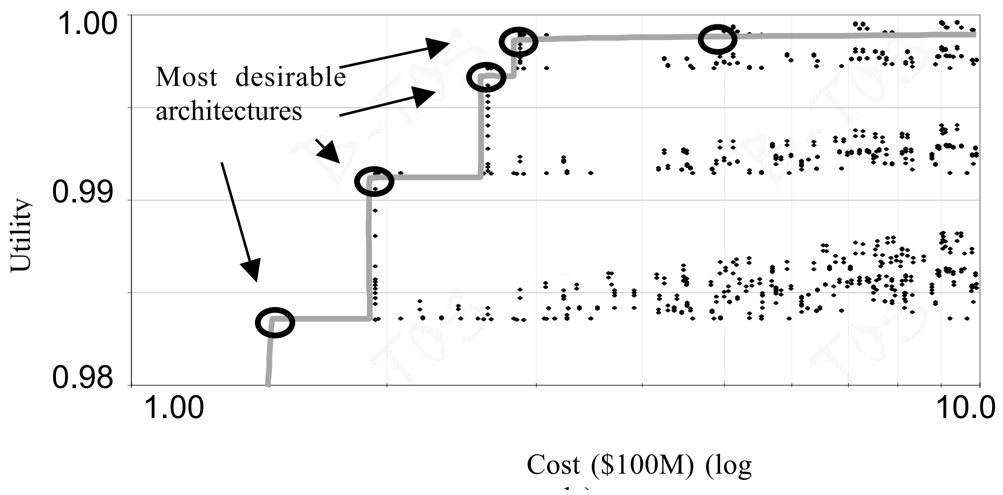


Figure 9-18 B-TOS Tradespace showing stair-stepped Pareto front.

NOTES AND REFERENCES

[1] Diller, N. P., "Utilizing Multiple Attribute Tradespace Exploration with Concurrent Design for Creating Aerospace Systems Requirements," Master of Science Thesis in Aeronautics and Astronautics, Massachusetts Institute of Technology, June 2002.

[2] Ross, A. M., "Multi-Attribute Tradespace Exploration with Concurrent Design as a Value-Centric Framework for Space System Architecture and Design," Master of Science Thesis in Aeronautics and Astronautics, Massachusetts Institute of Technology, June 2003.

[3] Much of this section is taken from an unpublished report by Nirav Shah

[4] Text in this section modified from Hugh McManus, *Product Development Value Stream Mapping, Beta Release*, Lean Aerospace Initiative, MIT, Cambridge MA, March 2004, p. 52.

[5] See SMAD, Chapters 10-18.

[6] McManus, H. L. and Schuman, T. E., "Understanding the Orbital Transfer Vehicle Trade Space," AIAA Paper 2003-6370, Sept. 2003.

[7] Stump, G., Simpson, T. W., Yukish, M., and Bennett, L., "Multidimensional Visualization and its Application to a Design-by-Shopping Paradigm," AIAA Paper 2002-5622, 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, Sept. 2002.

[8] This section is an edited version of section 6 of "X-TOS: 16.89 Final Design Report," Aeronautics and Astronautics Department, Massachusetts Institute of Technology, May 2002.

[9] Figure from Ross, A. M., Diller, N. P., Hastings, D. E., and Warmkessel, J. M., "Multi-Attribute Tradespace Exploration as a Front-End for Effective Space System Design," *Journal of Spacecraft and Rockets*, January 2004.

[10] Much of this section is taken, only lightly edited, from McManus, H. L. and Schuman, T. E., "Understanding the Orbital Transfer Vehicle Trade Space," AIAA Paper 2003-6370, Sept. 2003.

[11] Modified from Hugh L. McManus, Daniel E. Hastings, and Joyce M. Warmkessel, "New Methods for Rapid Architecture Selection and Conceptual Design," accepted for publication in the *Journal of Spacecraft and Rocket,* Jan. 2004.

[12] Jilla, C. D., "A Multiobjective, Multidisciplinary Design Optimization Methodology for the Conceptual Design of Distributed Satellite Systems," Doctoral Thesis in Aeronautics and Astronautics, Massachusetts Institute of Technology, May 2002.

[13] From Jilla, above.

[14] From Jilla, above.

[15] Warmkessel, J. M., and Kaliardos, W. N., "Architecting Space Systems," Proceedings of the 11th Annual International Symposium of the International Council On Systems Engineering (INCOSE), Melboune, Australia, July 2001.

---

16 McManus, H. L., and Warmkessel, J. M., "Creating Advanced Architectures for Space Systems: Emergent Lessons from New Processes," *Journal of Spacecraft and Rockets*, January/February 2004.

17 Walton, M. A, and Hastings, D. E., "Applications of Uncertainty Analysis Applied to Architecture Selection of Satellite Systems" *Journal of Spacecraft and Rockets,* January/February 2004.

18 Weigel, A. L., and Hastings, D. E., "Measuring the Value of Designing for Uncertain Future Downward Budget Instabilities," *Journal of Spacecraft and Rockets*, January/February 2004.

19 McManus, H. L., Warmkessel, J. M., and Kaliardos, W. N., "An Educational Experience in Developing Conceptual Architectures for Complex Systems," Proceedings of the 32nd ASEE/IEEE Frontiers in Education Conference, Boston, MA, November, 2002, pp. 361-366.