Tailored systems engineering processes for low cost high risk missions.

Jared Clements and Tyler Murphy
ATA Aerospace
1851 Charlene Dr. Kirtland AFB, NM 87117

Lee Jasper
Space Dynamics Laboratory
1851 Charlene Dr. Kirtland AFB, NM 87117

Charlene Jacka
AFRL/RVEN
1851 Charlene Dr. Kirtland AFB, NM 87117

ABSTRACT

Given the low cost of most cubesat missions, a full implementation of the traditional space systems engineering process to cubesat missions can be detrimental to programmatic success of the cubesat. At the other extreme, cubesat missions often suffer predictable consequences from the omission of standard systems engineering processes such as risk management, configuration management, and quality assurance. In this paper we discuss a scaled systems engineering approach to cubesat missions implemented on a programmatically constrained mission. A discussion of each of the standard systems engineering processes and options for tailoring the processes for a constraint-based mission and how this varies from the typical top-down mission processes. The intent is to inform the decisions of mission developers in determining what level of rigor is appropriate for each process in their unique circumstances and mission needs. Examples of tailoring processes utilized with missions currently underway at the Air Force Research Laboratory's Small Satellite Branch (AFRL/RVEN) are used to illustrate the application of the information presented.

CONSTRAINT-DRIVEN DESIGN

Small satellites are seeing significant utilization because they are intended to be both lower cost and more rapidly deployed; these attributes allow for a much wider range of people and organizations to build spacecraft. While small satellite platforms are not nearly as capable as their larger, more 'traditional' counterparts, they are facilitating large growth and investment. Since just 2015 well over 600 CubeSats have flown [1, 2] and it is expected that much greater adoption of the small satellite form factors will continue with investments on the order of tens of billions of dollars [3, 4]. The schedule and cost savings appear, so far, to have justified the reduced capability imposed by this smaller form factor.

With the growing interest, and investment, in these platforms there is a growing level of scrutiny being applied to the small satellite industry. Common space industry practices are being applied to small satellites that have been developed for larger one-of-a-kind space assets [5, 6]. Essentially, many organizations are attempting to develop small satellites to Class D or (the ambiguous) sub-Class D level of system engineering and mission assurance.

While Class D missions can be applicable to any size of space system, the reality is that small satellites generally do not meet the intent of Class D. The growing prevalence of small satellites are also starting to violate the assumptions Class D was predicated on: that these are one-of-a-kind. Class D is a higher risk posture but has evolved (or always was) assuming a relatively high probability of mission success. The small satellite community, and the design principles therein, have evolved from the concept of pushing the boundary on faster innovation. The small satellite community's innovation cycle was enabled by the community adoption of the containerized 1U standard. This standard has since been adapted to larger form factors but the fundamental design trades were developed in a form factor that were amenable for wide spread adoption. This standard has allowed the community to focus on innovation in processes and platform capabilities atypical of larger scale missions.

Further, these systems are greatly constrained and often are not capable of achieving something like Class D. The form factor imposes many physics-based limitations (volume, mass, power), many technologies are relatively low Technical Readiness Level (TRL), and the greater space industry holds many misperceptions about these vehicles (e.g. 50% of all small satellites are dead on arrival to orbit; the actual number is more like 17% [1, 2]). Because of the perception that these spacecraft are cheaper and faster, their schedules and budgets are often more static than the traditional "big space" paradigm. This drives capability, system engineering processes, and mission assurance.

It is recognized within the small satellite community that high levels of system engineering and mission assurance processes can reduce the innovative intention of small satellites. Where possible, the idea that a small satellite mission will "fit the box" instead of "building the box" has been utilized to help scope missions implemented in a small satellite form factor, as shown in Figure 1. While these ideas have been in the small satellite community for years, they have only recently been more directly discussed [7, 8, 9].

Constraint-driven design is where schedule, cost, and existing limitations (both technical and policy) drive the mission scope and execution plan. This is, so far, how most small satellite platforms have been designed and is in contrast to the "big space" requirements-driven paradigm. *Requirements-driven design* prioritizes mission scope over schedule, cost, or other limitations that may drive larger development efforts.

In order to be constraint-driven and reap the benefits of faster and cheaper, a mission's scope must be well defined and limited [8] or the scope must be flexible to reductions as constraints are realized. This idea can be challenging and even abrasive to much of "big space" but it is familiar to many small satellite crafters [2]. Assuming this step can be taken with mission stakeholders, the next most important attribute to a constraint-driven mission is scaling the systems engineering practices: the focus of this paper.

It is a common refrain for those working on small satellites that certain practices are not conducted "because it's a SmallSat". This is, in of itself, not sufficient or technically correct. Small satellites go through all of the same phases and steps as any space vehicle however there are many practices and processes that are either done on a very small scale or not applicable. The processes also tend to be iterative versus serial, with smaller scale processes happening throughout the mission lifecycle. Tailoring of these practices and processes to be constraint-driven is discussed and recommendations are made based upon experience from various AFRL programs and the University NanoSatellite Program. Further, discussion of good practices for improving resilience/robustness of space vehicles, without necessarily increasing system engineering or mission assurance burden, are discussed.

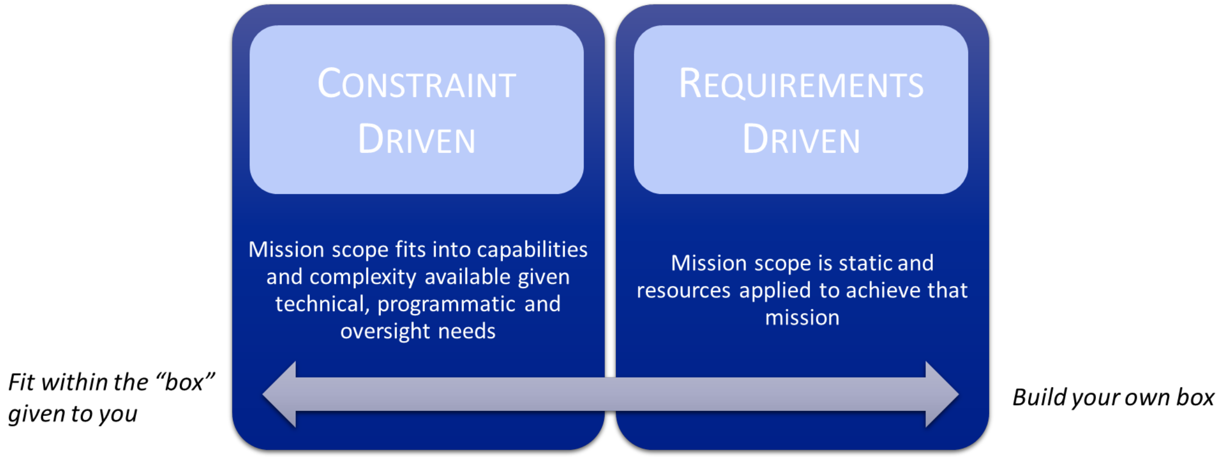
Constraint Driven
Mission scope fits into capabilities and complexity available given technical, programmatic and oversight needs
Requirements Driven
Mission scope is static and resources applied to achieve that mission
Fit within the "box" given to you
Build your own box

Figure 1: Constraint vs. Requirements driven missions [7]

VITALITY OF MISSION SCOPE

Unlike the wide and deep requirements of the traditional spacecraft development approach the requirements in a constraint driven model are kept at a high level and focused on the specific capability that is required to be demonstrated on orbit. The scope should cover the overall definition of what the mission is supposed to accomplish and a specific description of what the end result should be. Detailed (or deep) requirements are still necessary but they are only created when they are needed. It is important through this process to not over-define the solution space but rather the problem that needs to be solved.

The key piece of information here that drives scope is the Minimum Viable Product which is tied directly to the capability that should be validated on-orbit. Each capability has at least one on orbit demonstration associated with it for on-orbit validation purposes. Note that if there is not an on-obit test associated with the capability then the associated development is descoped from the mission. This scoping effort drives many of the systems engineering design trades that are determined through a mission lifecycle and fundamentally bound the programmatic constraints of the mission.

In constraint driven models the scope of a mission is controlled not fixed. It is expected that the scope will change throughout the mission lifecycle; this change is documented throughout mission lifecycle at programmatic reviews. It is critical that programmatic discipline is maintained to only add capabilities when they have made space by removing other capabilities first. Scope creep, where mission stakeholders add desired capabilities outside the necessity of the Minimum Viable Product, is a real danger to the success of a mission.

It is critical to document exactly how and when a mission's objectives are to be achieved by showing the major products, milestones, activities, and resources required for the mission. In traditional management the scope, cost and schedule imply high quality attributes which is locked down at the start of the project, conversely, in a constraint driven model the mission should deliver the desired scope, in the time allowed, within the budget allocated, and to the quality aspired to. The systems engineering processes tailoring therefore is a conversation between all stakeholders which is clearly defined at the beginning of a mission, so that mission expectations and programmatic constraints can be realized as early as possible in the mission lifecycle.

SYSTEMS ENGINEERING PROCESSES

Though there are several definitions of the various systems engineering processes in use today this paper will reference the IEEE 15288 definitions and process breakdown. Table 1 presents the processes that we will be discussing in this paper, broken down into Technical Management Processes and Technical Processes following the breakdown given in the DOD Best Practices for Using Systems Engineering Standards document [10]. Note that several of the processes called out in 15288 are considered out of scope for this paper, consisting of Acquisition, Supply, Life Cycle Model Management, Infrastructure Management, Portfolio Management, Human Resources Management, Quality Management, and Knowledge Management. Though critical to the success of an organization, this paper will be neglecting discussion of the larger processes and focusing on the processes that are within the scope of a single project.

Table 1. Systems engineering processes

Technical Management Processes	Technical Processes
• Project Planning • Project Assessment and Control • Decision Management • Risk Management • Information Management • Configuration Management • Quality Assurance • Measurement	• Mission Analysis • User Requirements Definition • System Requirements Definition • Architecture Definition • Design Definition • System Analysis • Implementation • Integration • Verification • Transition • Validation • Operation • Maintenance • Disposal

TECHNICAL MANAGEMENT PROCESSES

Accurate project planning is generally considered the most difficult of the tasks that systems engineers are assigned. One often quoted rule of thumb is to multiply your most accurate cost and schedule estimate by pi (3.14) to get a realistic estimate, or the constant e (2.72) if you're feeling optimistic. While there are always unknowns that will trip up any program plan, there needs to be a recognition that there are significant outside factors that drive this perception. One significant one is the inherent optimism that is required when making a program plan under competitive circumstances. A green-light schedule that assumes zero problems will always be unrealistic, especially under cost-plus contracting; Firm-fixed price contracting has a strong tendency to bring clear-eyed realism to cost and schedule discussions, with those most familiar with the challenges of the project able to inject their concerns into the planning process. This, in turn, forces difficult discussions significantly earlier in the program, requiring more realistic cost-benefit trades to be made at the user level, and helps temper unrealistic expectations from mission sponsors. Cost overruns are still a significant fact of life, but when constraints imposed on missions are rooted in reality and cancellation is more than a threat, but a valid option for a program, cost and schedule realism can become part of the organizational culture.

Generally, even the cheapest missions will still undergo the full review process that is inherent in the Project

Assessment and Control process. Tailoring is applied to the individual review, with a certain level of informality and relaxation of rigor to the requirements that are levied at each review. One critical piece that is shared between this and the Decision Management process is to push the decision making power as far down the organization as possible [7]. This has the effect of minimizing the need to bring the reviewers up to speed on the current state of the mission and allows the review to focus on the current issues that need addressing before moving forward. Continuity of management (driven by short schedules) also helps this process drastically, maintaining familiarity with the mission and knowledge of the previous decisions.

Risk management is generally one area where process is tailored generally falls to an identification of the primary risks at every review, with appropriate mitigation as it relates to the mission success. For many missions, large risk items that would be unacceptable for higher class missions are routinely accepted, such as the use of industrial quality electronics and unknown radiation susceptibility (generally a community practice). Mitigating the lack of more structured risk management is the smaller teams that are enforced by the low budgets of these missions. The improved communication amongst the small team allows the systems engineers to discover the risks inherent in specific courses of action. Also key is having the expertise available necessary to understand new found risks and mitigations quickly.

The adoption of new toolsets such as Confluence or other wiki-based systems has enabled significantly lower friction information management processes than predecessor file-based toolsets. Accompanying delegation down the organization structure of approval and review authority, as well as relaxation of some of the related formalisms also simplifies and speeds information transfer through the wiki-based toolsets.

Configuration management and quality assurance are often lumped together because of the overlap in both objectives and processes. A significant relaxation that is applied is the ability to work both tests and assembly procedures without detailed procedures. When the test requirements and test flow has been discussed with the appropriate approvers the test can be run and documented live, providing a significant speedup. Integration with the wiki-based information management system has also improved the ability to capture critical information from the procedure. Some relaxation of the standard two person rule has been tolerated, mostly in relaxing the knowledge requirements of the second person, where a tech or engineer with unrelated expertise can review and sign off on an action with appropriate explanation by the acting person. Flight hardware handling practices include ESD safety, smocks, hairnets, gloves, and a class 10K clean environment.

Measurement processes are generally associated with tool location and calibration tracking. Poor calibration practices can come back to damage a spacecraft in the most inopportune times, giving little options to scale back calibration practices. In general, tools lost inside spacecraft hardware can be mission ending, but with spacecraft as small as these there are few opportunities to misplace tools.

For many of these systems engineering processes there is the recognition that while process can improve consistency, in can also reduce individual responsibility and ownership. Delegation of authority is critical to improve ownership and responsibility such that relaxation of process can reduce cost and schedule without catastrophic results.

TECHNICAL PROCESSES

The technical processes in Table 1 generally follow a mission flow, with the exception of the System Analysis process, which is cross cutting throughout the mission. Figure 2 shows the connection between the processes and the mission lifecycle for a satellite mission.

Concept development

The early stages of mission lifecycle are likely the least well defined. The goals of the early stages of mission development is to identify a self consistent set of mission objectives, requirements, and architecture that are feasible within cost and schedule constraints. Sometimes this is straight forward, such as when a customer approaches with a well scoped component test idea. Usually there will be several iterations of concept development, including cost and schedule estimates, returning to the customer to discuss options and possibilities, before a committment is made.

Concept development generally consists of rapid iterations on the systems budgets, such as communications, power, pointing, navigation, etc., evaluating changes to the mission and experiment CONOPS enabled by various options. Impacts to the requirement set and system architecture guide new decisions. Key performance parameters drive decisions and guide the selection process.

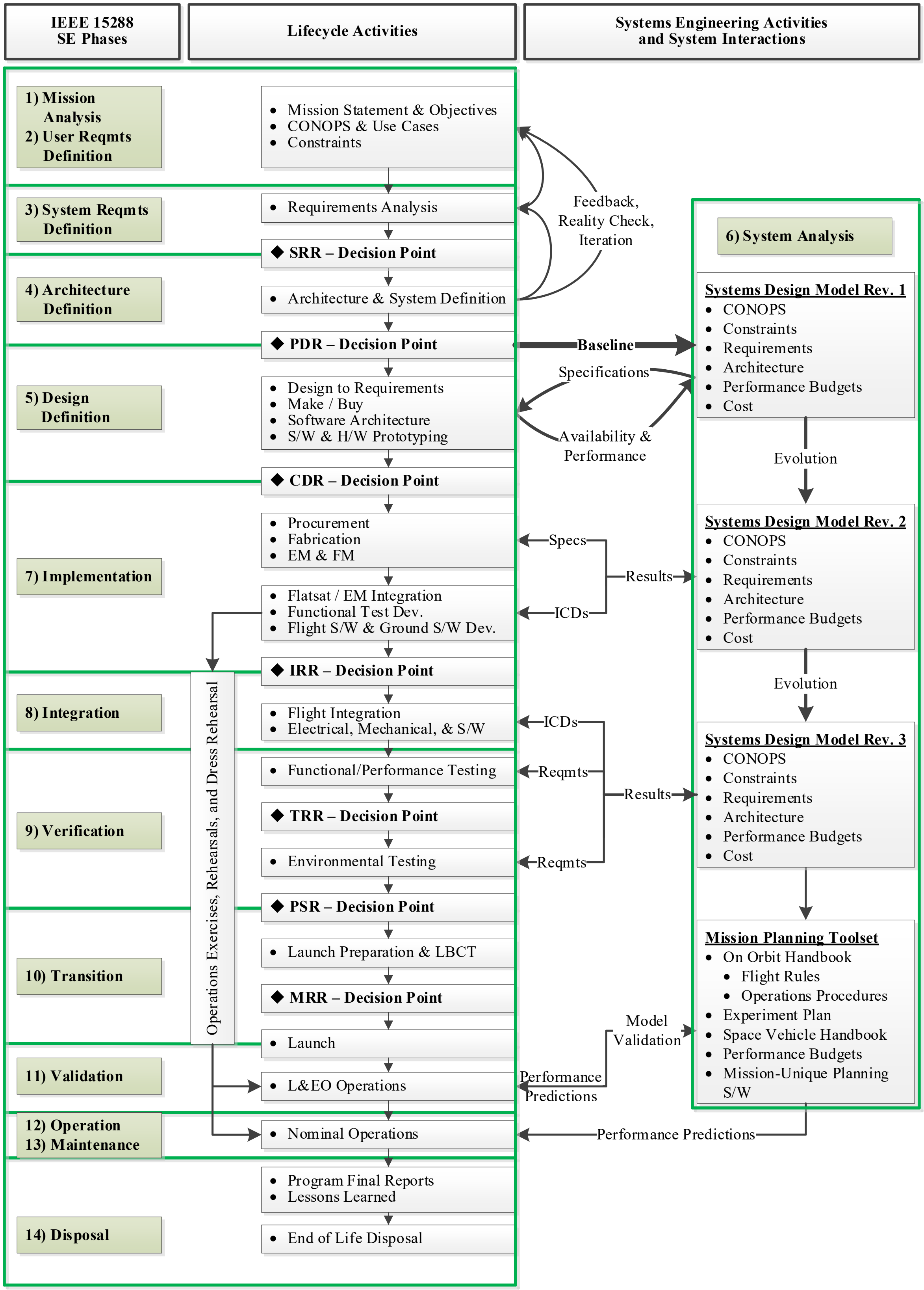

Figure 2. Systems engineering processes and the mission lifecycle.

The baseline for the system is implemented and documented in a system design model which consists of the CONOPS, requirements and constraints with functional, performance, and environmental testing defined, architecture, and the performance and cost budgets. Further detail is required in a risk assessment and mitigation plan. The generation of a self consistent system design model is necessary to progress to PDR.

This top level description of the concept development process most likely applies across all mission classes, the key ideas that change with mission class is that process is looking for a well scoped minimum viable product that there is reasonable agreement is worthwhile to embark on for the cost and schedule resources available. This can be low risk, such as a widget testing mission, or high risk, such as attempting to interface with a global satcom constellation that

Design definition

Between PDR and CDR the design is fleshed out through procuring or developing the subsystems and components that meet the detailed requirements. One simplification applied is a strong preference towards buy in make/buy decisions. Full design rigor is generally expected when the decision is to make the component in house.

One simplification to the review and approval process that can be adopted is a peer technical review, which is a detailed, but often informal, assessment of the work conducted on a component, subsystem, system, etc. The intent is to get a second set of eyes to better catch errors, omissions of best practices, cross pollinate ideas, and to provide more cross-team communication. This review may come from a subject matter expert or similarly skilled engineer from another project.

In many cases it is quicker and cheaper to begin prototyping early in this process, allowing the engineers to evaluate design and component selection decisions while providing time to correct mistakes. The increasing complexity the various ICs available today increases the challenge of catching errors at the schematic level, often the only way to determine if a chip can perform the required task is to prototype the circuit and work out the proper settings by hand.

Canonically, software development work prior to CDR should be limited to architecture, prototyping, and planning. However, most missions can attest to the wisdom of an early start to the software development. In this case, the use of non-EEE parts can significantly enhance the capability of the processing on the spacecraft, and has enabled significant sophistication in the flight software of cubesat missions. At the same time, if mission scope can be reduced sufficiently the mission logic may be able to fit in basic microcontrollers, significantly reducing the time and financial investment required for the software development.

Certain judicial enhancements to the mission at this design stage can minimize cost and personnel commitments during both testing and operations. One requirement that is generally carried on AFRL/RVEN missions is to be power positive in a tumble. This requirement enables the critical components to be reduced to the power, TT&C, and connecting subsystems (usually command and data handling). The elimination of the attitude determination and control subsystem from the system safe mode allows for both simplified operations (e.g. business hours only, progressing to unattended operations), and a reduction in testing in the ADCS system, due to the knowledge that it is not a critical subsystem.

System analysis

The system design model is the central analysis tool that supports the systems analysis portion of the systems engineering process. The model captures the mission and experiment CONOPS, the requirements flowdown, product break down and work break down structure, and the ICDs.

The interaction between the system design model and the system changes throughout the mission lifetime. Most of the design work on the mission occurs in the system design model prior to PDR. Between PDR and CDR the model is updated to reflect component availability and feasibility, cost benefit analyses, design trades, and evolving schedule and cost constraints. The final CONOPS scenarios, design, and expected performance are captured at CDR.

After CDR the model serves as the basis for defining test campaigns and incorporating test results into performance predictions. The model informs flatsat, hardware-in-the-loop and software-in-the-loop testing and provides the proper location to incorporate the record as-built performance and calculate system margins and capability. It also provides the ability to analyze the impact of a failed test and informs the decision to modify the design, modify the test, or accept it as is with a waiver.

In AI&T, the model helps specify the functional and environmental testing to ensure a test-as-you-fly approach. As final testing wraps up the model is used to develop operations plans and a mission planning toolset for use during early, nominal and contingency operations.

Implementation

The functions of implementation are the procurement and fabrication of the various parts, components, and subsystems. One particularly powerful simplification of this process in the use of a flatsat, where non-flight boards and harnesses are electrically integrated in a tabletop setting. This encourages rapid identification and correction of flaws in design, ICD mismatches, and most non-mechanical issues. The flatsat allows for breaking connections and break-out box level verification of key measurements that are infeasible after mechanical integration

The flatsat also allows for early functional test development, which provides time for iteration on the functional test procedures and helps in catching design flaws, allowing for later flight hardware functional tests to only focus on workmanship flaws. A heavy focus is placed on test scripting.

The flatsat also provides an ideal platform for flight software testing. The acceleration of flight software development on the flatsat is likely sufficient justification for the apparent extra effort even without the other advantages described here.

Assembly, Integration, & Testing

The AI&T phase of any mission can make or break both a mission's schedule and budget. During this phase of mission development, many of the investments or shot comings made in the earlier missing phases are realized.

Traditional mission AI&T focuses heavily on the carefully developed integration procedures with multiple levels of inspection and may even include the construction of an engineering unit to test these procedures. These practices, while well suited for Requirements-driven missions, significantly increase both the cost and schedule for the mission. For Constraint-driven missions, similar levels of mission assurance can be achieved through the application of some simple design practices and lean integration processes specifically applied to mission critical integration activities.

In general, small satellite missions are designed and built by much smaller teams than their traditional counterparts. This allows the design team to also act as the AI&T team. Having these functions so closely coupled allows the AI&T team to become experts with their system during the design and since they don't handoff AI&T to a separate team, there is less need to meticulously design integration procedures. With this level of understanding of the design intent, procedures can focus on critical integration activities, such as optical alignments, and less on the integration of more robust systems.

The testing and verification of constraint-driven missions also varies significantly from the traditional paradigm. While the same objectives of verifying that the system will survive launch and perform the mission objective still apply, the level to which this verification is performed is where constrain-driven missions vary the most. For these missions, it has been found the that the greatest return on investment comes from the following basic tests:

Functional Day-in-the-Life (DITL)

DITL testing, when properly designed, should accurately demonstrate the critical functionality of the spacecraft. This usually focuses first on initial startup and system checkout and then exercises the operational modes. Some simple error detection and recovery testing may be performed but it is not the intent of constraint-driven DITL to exercise all of the edge cases but to simply verify that the system performs as intended. This test specifically includes the launch and early operations sequence.

Power Characterization

As the power subsystem represents the lifeblood of the spacecraft, significant efforts are expended to verify the full functionality of the subsystem. This includes verification of the depth of discharge, recharge through solar panels, autonomous recognition of safety limits on the battery, proper inhibit functionality, load testing and switching, and proper telemetry production.

Long Range Communications Verification

Small satellite systems present a unique opportunity to test a full end to end communications path of the satellite that simply could not be performed with larger systems. Due to their size, the satellite can either be tested by free air radiating with a significant distance between the test antenna and the satellite or even with an actual ground station asset. It has been found that many issues can be discovered by performing a long range test that would otherwise be missed when using either an antenna hat or performing attenuated hardline tests.

Command and Execution Test

Full verification of the software functionality is required, though there is some flexibility on whether that is performed on the flatsat, flight vehicle, or a simulator. This is an execution of each command in the Command and Telemetry List (CTL). The depth to which all the various permutations of arguments for each command is verified is allowed to fluctuate depending on the mission.

Full Functional Test

Functional testing on the balance of the subsystems is allowed to stay at a high level, emulating the expected use cases that each component may see in operations. If failures are encountered further investigation is required. Often there are edge and corner cases that are not well explored or tested, and these can be discovered on orbit. The expectation is that as long as the critical subsystems are well characterized these faults are recoverable and can be dealt with during operations.

CG/MOI Testing and Polarity Checks

These tests gather the required information to ensure that the ADCS system and algorithms are provided with the most accurate information. The polarity checks also ensure that the sensors and actuators were installed correctly.

Other tests that may be performed, given the specific risk tolerance posture of the mission, these include EMI/EMC testing, detailed ADCS testing, and payload performance testing.

Vibration Testing

More traditional systems may test all components independently prior to integration and modeling the integrated system prior to full vehicle vibration testing. Constraint-driven missions can realize significant cost and schedule savings by only vibration testing the fully integrated system and limiting modal modeling to only extremely sensitive components.

Thermal Vacuum Testing

Testing the system under both hot and cold vacuum ensures that the system will perform as designed on orbit. While the duration and number of cycles can vary from mission to mission, limiting the number of cycles can significantly reduce the cost and schedule.

MISSION OPERATIONS

Traditional mission operations consist of several operators sending command sets up to the spacecraft in a serial process. This method of controlling is well suited for the requirements-driven mission as it provides a man in the loop to ensure that the spacecraft remains operational as much as possible.

For a constraint-driven mission this operations paradigm must also be adjusted. Many of these missions have much more constrained operations budgets that drive a push to operate as "lights out" as possible. "Lights out" operation is a method of operating a spacecraft with either very limited or actually zero controllers sitting in the mission control center.

This operations method is achieved through the careful design of the constraint-driven system to include two design principles. The first of these is a tumble proof COM link. By providing a communications link that can still close the link with the ground even in a tumble, operators can recover the vehicle from anomalies much quicker as well as monitor the system state of health even in if it currently is unable to recover from a current power condition. Once the power system recovers, operators can then proceed with bringing the system back online.

The second operations enabling principle is to utilize the DITL testing to develop mission operations scripting. By developing and utilizing this scripting during the testing phase, these command sets can be "canned" and used for future operations. By designing in this way, operations planning can then be accomplished during a weekly planning meeting rather than the more traditional daily planning.

CONCLUSIONS

The majority of space organizations have evolved to be requirements driven such that meeting mission goals, and scope, take a level of precedence over cost and schedule due to the limited access to space. However, as access to space continues to expand for small satellites, and the need for rapid capability development increases, schedule and cost are driving mission lifecycles. These Constraint-Based missions require tailored systems engineering practices that prioritize demonstrated capability with a lower performance over undemonstrated capability with higher performance. The small satellite community should adopt a process that verifies mission success allowing the mission validation to occur on orbit allowing rapid demonstration of capability.

References

1. Swartwout M., "CubeSat Database," Saint Louis University, 2019. https://sites.google.com/a/slu.edu/swartwout/home/cubesat-database
2. Swartwout, M., "CubeSat Mission Success: Are We Getting Better?," Proceedings of the CubeSat Developers' Workshop, CalPoly, 23 April 2019.
3. "Global Prospects for the Small Satellite Market, 2018-2022," ResearchAndMarkets.com, 27 March 2019.
4. "Global Small Satellite Market Insights, Forecast to 2025," The Market Reports, January 2019. Report Code: 1237587.
5. "Design, Construction, and Testing Requirements for one of a kind space equipment," SPVT-2016-

005, ORIGINAL ED., DOD-HDBK-343. February 1986.

6. Johnson-Roth, G., "Mission Assurance Guidelines for A-D Mission Risk Classes", Aerospace Corporation, TOR-2011(8591)-21, June 2011.

7. Jasper, L. E. Z. and L. Hunt and D. Voss and C. Jacka, "Defining a New Mission Assurance Philosophy for Small Satellites," SmallSat Conference, Logan, UT, Aug 4-9, 2018. Paper No. SSC18-WKII-05

8. Tolmasoff, M. and R.S. Delos and C. Venturini, "Improving Mission Success of CubeSats," Proceedings of the U.S. Space Program Mission Assurance Improvement Workshop, The Boeing Company, El Segundo, CA, June 2017.

9. Johnson, M. A. and P. Beauchamp and H. Schone and C. Venturini and L. E. Z. Jasper and R. Robertson and M. Moe and J. Leitner and F. Tan, "The Small Satellite Reliability Initiative: A Public-Private Effort Addressing SmallSat Mission Confidence," SmallSat Conference, Logan, UT, Aug 4-9, 2018. Paper No. SSC18-IV-01

10. "Best Practices for using Systems Engineering Standards (ISO/IEC/IEEE 15288, IEEE 15288.1 and IEEE 15288.2) on Contracts for Department of Defense Acquisition Programs," Office of the Deputy Assistant Secretary of Defense, April 2017, http://www.acq.osd.mil/se/