Doing Things Right in Space Programs

Article #4

Common Problems in Developing a Space or Launch System

Part 2

April, 2000

Tom Sarafin

President, Instar Engineering and Consulting, Inc.

6901 S. Pierce St., Suite 384, Littleton, CO 80128 • (303) 973-2316 • instarengineering.com

In last month's article, I addressed some of the problems that hit space programs over and over again. Here are some more:

- **Predicted launch loads going up after the hardware is built**

Nearly every spacecraft program in which I've been involved has had problems with predicted launch loads. Just before launch, the latest predicted loads for something somewhere in the vehicle exceed the loads to which it was designed and tested. Typically the new loads come from analysis done with test-correlated models, which, of course, we can't have until we've committed to the design and built the structure. Often the higher loads lead to negative structural margins of safety, which in turn drive costly redesign or retest, or additional risk. Wherever I teach throughout the industry, I hear the same problem. Apparently, we have accepted it as a necessary evil.

I can't imagine any other industry putting up with this problem: spending jillions of dollars designing, building, and testing to loads and environments that appear to come from a roulette wheel.

Much of the problem is because of our process, along with the accompanying mindset. We predict loads and react to them when we should be trying to control them. Spacecraft programs have engineers whose job is to control temperatures in space; can you imagine if, instead, they simply predicted temperatures and had the rest of us react to their predictions? That's exactly what we do with launch loads. What's wrong with this picture?

Designing to control launch loads for a payload requires knowledgeable engineers, cooperation between launch vehicle (LV) and payload organizations, effective tools, and enough time to make it happen. First we must understand the environments and the structural characteristics of the launch vehicle (knowledge obtained through cooperation). Next, through sensitivity studies, we identify desirable features for our payload, such as target natural frequencies. Then, through more studies, we identify how sensitive the loads are to the design parameters we can't control. Finally, we make our design robust, or tolerant of the uncertainty.

On most programs, the loads engineers want to do these things, but they don't have enough time. Part of the blame lies in cutting design budgets and schedules, but not all. With the current process, the payload contractor waits three to six months to see the results of a single iteration, or *loads cycle*. Such a loads cycle includes *coupled loads analysis* (CLA), in which the payload and LV math models are combined. CLA is typically done by the LV organization.

No program can do enough iterations to develop robust structural designs when a single iteration takes half a year. Only through simplification—designs, math models, software—and teamwork can we make iterations feasible. Unfortunately, as our engineers have become more and more specialized, with ever narrowing blinders to the overall project, designs have become more complex, and math models nowadays are enormous. Model size has always been constrained by computers, but the constraints are much broader now. There's much more space to fill, and we are happy to fill it. Software tools that enable seamless generation of finite-element models and thermal models from solid-model design databases actually encourage complexity. Developing a simple model—as well as finding a simple design—takes more time, which is something our engineers haven't had much of lately.

I need to be fair, though. I've been awfully critical of management cutting budgets and compressing schedules. When I read what I have written above, I almost convince myself that, with enough education and invested time, it's easy to avoid loads problems while developing an LV payload. It's not. This is a difficult problem, as I addressed in my second article. We won't know the launch loads until we build something and launch it, and they won't be the same the next launch. Simplifying models, improving the loads-cycle process, and doing sensitivity studies will help, but they won't make all our problems disappear.

There is another, related approach that may solve much of the problem: isolating the payload from the launch vehicle. Every time we plan to put a different payload in a car or a truck, we don't first have to do a coupled loads analysis. Why? Because someone engineered the vehicle to isolate the payload from the excitation of tires on a bumpy road. If we were to hard-mount ourselves to the axle of our car, the way we do with a spacecraft to a vibrating launch vehicle, we would feel every bump in the road, and our teeth might shatter. Instead, our car has springs that keep us from feeling much high-frequency vibration and shocks to damp out any low-frequency vibration.

We can take the same approach with mounting payloads in launch vehicles, but it will take a change in mindset. The biggest argument against isolating payloads from launch vehicles is that the flexible springs needed to do the job would allow the payload to bang into the payload fairing when loaded laterally. There simply is not enough clearance between the two structures.

But let's look closer: the reason there is not enough clearance is because we filled the available space. Programs tend not to get serious about isolating their payloads until they have problems with predicted loads, and by then it's often too late. We can isolate payloads—and some programs are doing so—but we can do it cost-effectively only if we plan on it from the beginning. If we leave more clearance when configuring our spacecraft, we'll have the room needed to make isolation feasible. And if we make our isolation system highly damped (even though that would reduce the effectiveness of isolation), we should be able to keep any low-frequency vibration from confusing the vehicle's control system.

I am still looking for the launch-vehicle program that is seriously interested in breaking the mold and providing a better service to their customers with a standard mounting adapter that includes a tunable isolation system. The first company that succeeds in this endeavor will knock the socks off its competition as payload developers put more emphasis on the indirect costs of LV options.

- **Excessive design and test environments, driving weight and test failures**

Depending on your position in the space industry, you may or may not be aware of this problem. We generally consider weight (or mass) to be critical in our industry. Programs have willingly spent many thousands of dollars to reduce weight by a single pound. Despite this weight criticality, the loads we use for design and test are often many times higher than what the hardware will see during the mission. Quite often, this is the case because of the problem discussed above: predicted loads going up after everything is built. Our natural tendency is to pad our early predictions the next time around so it won't happen again. But there are other reasons as well that are tied to a lack of data, a lack of understanding by the engineering team, and a misconception that we have to do tests a certain way. This is an area in which practical education and a willingness to challenge the traditional way of thinking can pay great dividends.

Did you know that a traditional random vibration test often stresses the materials in the structural housing of a component (black box, reaction wheel, etc.) ten or more times as severely as does launch? The same is true for propellant lines and component support structures; and the overtest becomes even more severe with traditional random vibration tests of larger items, such as subsystems and full spacecraft. Sinusoidal vibration testing can be even more excessive.

The Jet Propulsion Laboratory has led research over the past several years to change the way we do vibration tests with a technique called *force limiting*. With this approach, the test equipment monitors and controls not only the acceleration of the mounting fixture but also the force transferred between the fixture and the test article. Doing so achieves a much more realistic test and thus, because we can design to less load, reduces the weight of our components. Over the past ten years or so, many programs have taken advantage of force-limited vibration testing, but many others have not—for several possible reasons:

- a lack of understanding (too busy to become better educated? no budget for continuing education?)
- an unwillingness to invest in new technology (yes, special equipment is needed) or to accept a new way of thinking
- or perhaps a declining interest in R&D as a result of year after year of having no overhead money available for it

Loads engineers, as is the case with most engineers, always want to be on the safe side. This is a healthy attitude, but when each engineer in a serial process is unaware of the conservative assumptions the others are making, designs are overly penalized.

In one of my classes, several of my students told me the launch-vehicle program they were supporting had recently undergone a $3,000,000 change, in which they redesigned and replaced virtually all of the structural brackets and fittings that support the LV's components. The change was driven by a decision to increase the design loads from 50 g's to 100 g's. I don't know if the increase was because of ignorance, because of a lack of communication, or because the loads analysts didn't trust the stress analysts. But I do know that the entire mass of a sizable component will see nothing close to 100-g acceleration during launch.

Quite often, the reason design and test loads are excessive is that the loads engineers simply don't have time to strip the conservatism. Recall the example I gave in Article 1 of the predicted loads in a spacecraft going up by a factor of three because the loads engineer was overworked and had to make simplifying assumptions. It only stands to reason that, if we have continual problems with launch loads, the worst thing we could do on the next program is cut the budget for the loads analysis.

Table 4-1 summarizes common viewpoints from payload and LV developers regarding the above two problems.

Table 4-1. Why Do We Keep Having Problems with Launch Loads and Test Environments? Several things are at fault here. We need to improve our understanding of the problem, educate the people involved, and make sure they have enough time and resources to do a good job. We also must break down barriers between LV and spacecraft engineers so they can work more closely together and share any needed information. Finally, as I'll discuss in later articles, we should strive early in the program to design a mounting system that will decouple or isolate the spacecraft from the LV.

<table>
<tr><td colspan="2">Problems:
• Predicted launch loads going up after the hardware is built
• Excessive design and test environments, driving weight and test failures
Causes:</td></tr>
<tr><td>Spacecraft developer's point of view:</td><td>LV organization's point of view</td></tr>
<tr><td>– Weight is too critical for us to use big uncertainty factors.
– You (LV) guys need to give us the right loads for design at the start of the program.
– You won't give us enough information up front.
– It takes you 6 months to do a coupled loads analysis (CLA), which doesn't support our design schedules. Why can't <u>we</u> do the CLA?
– The loads you gave us are unreasonable. You're too conservative!
– Next time, I guess we'll have to take the weight impact and design to really big uncertainty factors.</td><td>– Your spacecraft model changed; it's interacting differently with the LV.
– You wouldn't have this problem if you had designed the spacecraft with natural frequencies all above 70 Hz.
– It took us 3 of those 6 months to debug that huge math model you sent us.
– The LV math model and forcing functions are proprietary, and our lawyers won't let us give them out because of liability.
– We only have so much data, and we have to be on the safe side.
– Next time, I guess we'll just have to be even more conservative with the loads we give out for preliminary design</td></tr>
</table>

- **Key decisions based on wrong information because documentation doesn't agree with the actual configuration**

Example: Just before the first Tethered Satellite mission in 1992, the predicted launch loads increased as a result of changes to the finite-element model. Negative margins of safety drove the program to add structural reinforcement, which included a protruding stud. The engineering database had indicated the stud would not interfere with the deployment mechanism's operation, so the program decided not to repeat the costly deployment test. But the database was wrong; it did not include a part on the moving guide that had been added sometime earlier but still late in the program. During the mission, this part jammed against the protruding stud, and the tether, designed to extend 20 km, only went out about 250 meters.

If you think this is an isolated instance, you're mistaken. In 1984, an astronaut used the Manned Maneuvering Unit (MMU) to approach the nonfunctional Solar Max spacecraft, with the intention of repairing it. He was not successful. A small plastic pin in a thermal blanket prevented the MMU from docking to the spacecraft. The engineers who had developed the docking method had been unaware of the pin's presence.

A mission's success often depends on the accuracy and completeness of the information available to the engineers. *Configuration control* is all about ensuring such accuracy and completeness so we can make good decisions. At all times after the hardware is built, we should be able to recreate it precisely, solely from our documentation. We won't always have the luxury of even looking at the hardware, let alone measuring parts of it. This is obviously the case after launch, but even before then, flight hardware can become inaccessible to decision-making engineers.

Although the above examples are old, I believe the potential for losing control of the configuration is greater today, mostly as a result of cost-cutting and diluted responsibility. We have cut back on documentation and product inspections, and on oversight by cognizant, responsible engineers. As I've pointed out in this series of articles, attempts to reduce cost by cutting activities that add quality are misguided and destined to fail.

On April 9, 1999, a Titan 4B launch vehicle put its payload in the wrong orbit because the first and second stages of the Boeing Inertial Upper Stage (IUS) failed to separate properly. Investigators concluded the problem was that the technicians, having properly followed written procedures, installed thermal wrap around a separable electrical connector in such a way that inhibited separation (source: Space News, September 6, 1999). In other words, the documentation (procedure) was incorrect, and no one knowledgeable enough to detect the problem did so.

Documentation has been under attack for some time now, based on the belief that it adds little or no value for its expense. I really can't argue that much of our documentation is not cost-effective; I've seen many supporting examples. But it's not the idea of documentation that is bad; it's the quality of our documentation. Just because we don't do something well is no reason to stop doing it. I argue that effective documentation is essential, so we must learn how to do it better. More than that, though, we must acknowledge that our documentation will never be perfect. If it were, we could automate the integration of a space vehicle. Cognizant, responsible people remain the key to success.

- **Mission failure**

I don't know where to begin here. Our industry has really taken a blow lately. We've been deservedly crucified by the press. I know I'm not alone when I feel almost too embarrassed to tell my neighbors what I do for a living.

This is not the industry I thought I was entering twenty-one years ago.

We all know we have a problem retaining talented, knowledgeable engineers—but is it any wonder? Who in their right minds would choose to work for years on a project that has such a high chance of failure?

The answer is that many people will do so, but only if they believe that they are part of something big, that they are attempting grand things that have never before been done. Our failures have been in things we know how to do and have successfully done. Most have occurred because we rushed the design, because we failed to communicate or pass on knowledge, because we couldn't afford thorough inspections and tests, or because we accepted risks that we didn't really understand.

I have been fortunate; all the space programs on which I've spent significant time have succeeded. (I do not mean to take credit, here.) I can only imagine the feeling of having something fail that I've adopted as my own and worked so passionately on. How would I accept the failure, knowing we could have prevented it by doing things right? How would I accept assignment to another program that is headed down the same path? I'm sure my passion would be gone.

Without passion on the part of the people involved, the space industry is doomed. The key to success—not just for the mission, but also in terms of containing cost—is our people. If morale is low at your organization, you will have high cost and many failures. Any fix you find that doesn't improve morale will not work. This is true of any business, actually, but I think you'd be hard pressed to find an industry in which overall morale is lower. And that's a shame. This should be the coolest job in the world!

If we're serious about turning things around in the space industry, we need to stop addressing the symptoms of the problem and start addressing the roots. Every time a mission fails, a team of experts investigates to find the cause, and then the program takes corrective action. Although this is a fundamentally correct process, I'm becoming disturbed by the findings. It appears that, once we find the specific problem, we are content to correct only that problem and not the root cause, which is in our process.

In the above IUS example, the Space News report implies the team investigating the failure concluded the problem was that a written procedure was inadequate. I'm sure there is more to it than this condensed report contains, but I've seen this sort of thing enough to be cynical. We tend to focus on the specific problem and then correct it, and then change nothing in the process that allowed such a thing to happen. It's simple enough to correct the procedure for insulating the IUS connector, but the question screaming in my mind is this: why was it that a knowledgeable, responsible engineer didn't check the work and find the problem with the thermal wrap? We'll never solve the kinds of problems we've been having by merely writing better procedures. Someone has to be clearly responsible for making sure things are properly put together.

Part of our problem is having too many specialized engineers responsible for the pieces without anyone responsible for the whole. We tend to dilute responsibility. In response to a mission failure, we may start up a new organization with an obscure responsibility that is just another way of saying "quality assurance," even though we still have a Quality Assurance group. Pretty soon, no one feels responsible because they believe someone else will take care of it. What else are we to conclude when we hear that the reason the Titan 4B delivered the $800-million Milstar 2 satellite to a useless orbit on April 30, 1999, was a known data-input error that no one followed up on? The solution, I believe, is <u>not</u> to add another layer of oversight and checking.

As a friend of mine—a quality engineer—and I agreed a few months ago, if we really had an effective quality system, there would be no need for a Quality Assurance group, let alone two or three. Everyone on the program would be a Quality engineer, with responsibility clearly defined. If we did have a QA group, they would be systems engineers, people responsible for coordinating interfaces to ensure subsystems and systems would work.

Many of my fourteen years at Martin Marietta in Denver (now Lockheed Martin) were on a single program, in which we did many things wrong but never had a mission failure. This program assigned responsibility to a single, capable person for ensuring the entire space vehicle, which was quite large, was properly integrated and tested. We called him the "Vehicle PIE" (Product Integrity Engineer). There was no question of who was responsible for making sure fasteners were properly torqued, electrical connectors properly mated, or thermal tape properly applied. All of that was his. The only way he could manage this responsibility was to ensure everyone working on the vehicle was also knowledgeable and responsible. And morale was not a problem.

In next month's article, I'll summarize the common reasons for the problems discussed above and in my previous article, and we'll begin to look at solutions.

About the Author

Tom Sarafin has been involved in the space industry full time since 1979, at which time he graduated from The Ohio State University with a BS in civil engineering and took a job as a stress analyst at Martin Marietta Astronautics in Denver, Colorado. While at Martin, he was involved with design, analysis, verification planning, and testing on several spacecraft and launch vehicle programs. After contributing to the book *Space Mission Analysis and Design* [Larson and Wertz, editors, first edition published in 1991], he obtained management's support and funding at Martin Marietta for the development of a book on the interdisciplinary development of structures for space missions, and served as principal author and editor for 23 other authors. He left Martin Marietta in 1993 to complete this book, under the guidance of Dr. Wiley Larson at the U.S. Air Force Academy. The result of nearly four years work—*Spacecraft Structures and Mechanisms: From Concept to Launch*—was published in 1995 jointly by Microcosm, Inc., and Kluwer Academic Publishers.

In 1993, Mr. Sarafin formed his own company, Instar Engineering and Consulting, Inc. Once he finished his book, he began providing review and advice as a consultant to space programs. He also developed a short course based on his book and began teaching it throughout the industry. The course has been quite popular, and the business has grown. Now Instar offers a curriculum of courses taught by experienced engineers and continues to add to that curriculum.