Instar

Doing Things Right in Space Programs

This article is part of a series started in January, 2000. My intent is to share a philosophy and ideas for how to increase the chances of success in space missions while also reducing total cost. Once these articles are completed, I plan to assemble them into a book. Please send comments to me at Tom.Sarafin@instarengineering.com.

Ten Principles for Doing Things Right in Space Programs

1. **Adopt the right attitude**
2. **Invest in knowledge and understanding**
3. **Instill ownership and responsibility**
4. **Constantly seek ways to improve teamwork**
5. **Follow a sound engineering approach**
6. **Reduce total cost through good engineering**
7. **Keep everything as simple as possible**
8. **Establish an effective quality system that involves everyone**
9. **Be willing to accept risks, but only those you truly understand**
10. **Make sure everyone has enough time, resources, and freedom to do things right**

Article #14

Follow a Sound Engineering Approach

Part 3: Developing a Verification Plan

April, 2001

Revised June, 2001

Tom Sarafin

President, Instar Engineering and Consulting, Inc.

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

The twelfth article in this series gave an overview of verification within the context of the systems-engineering process. If you haven't read that article, please do it now. I plan to repeat only a little of it here.

Because most of us don't have a sound understanding of verification, we tend to rely on our customer to dictate much of it and on documents from past programs to help us do the rest. The results, as I've pointed out several times in past articles, are costly activities that add little value and key issues not being addressed. If we remember that we are simply trying to assure ourselves and our customer that our products will work as specified during the mission, we will have a much easier time planning verification.

To develop an effective verification plan, we need two things: (1) a thorough understanding of the product's requirements (see my last article) and (2) wisdom regarding engineering principles, by which I mean "how things work" and "what can go wrong." With sound engineering, our job is to root out anything that can go wrong and ensure its likelihood is acceptably small.

Verification is essentially the same as quality assurance. Depending on how you look at it, either one could be considered part of the other. In the space industry, quality is reflected by the probability that the mission will succeed. For high quality, we must

- properly define the requirements,
- design the item to meet those requirements,
- build the item to its design specifications,
- and control the product's configuration up until launch.

Verification overlaps all of these activities. My previous article (#13) discussed how to systematically identify and communicate requirements. Although it serves other roles within the systems-engineering process, requirements definition is an essential part of verification, at the highest level. At the other end of the above list, we'll need a controlled system for ensuring that, once we know the item is built properly, the item's configuration does not change or, if it does, that we know what those changes are. With *configuration control*, the objective is to know at all times every detail of the product's makeup—without having to look at or otherwise inspect it. This is so important because our product gets more difficult to inspect as it becomes part of an increasingly larger system up until launch, after which we probably will have no access to it. A full configuration-control system encompasses requirements definition, design, and manufacturing as well.

Mostly, though, what we think of as verification concerns the middle two activities in the above list: Have we properly designed and built the item? These are separate issues, as the above list indicates, and it's important that we understand the distinction. Table 14-1 describes how the issues (things that can go wrong) differ between design and manufacturing. The key difference in verification is that, when the design is in doubt, we need to verify it only once, whereas, when the concern is with manufacturing, we must verify quality for each build.

Verification Methods

Depending on the requirement, we can verify it by one or more of the following:

– Analysis
– Process control

– Inspection
– Test

Another method you will hear quoted is *demonstration*, but in my mind, a demonstration is simply a form of test. Any test, including a demonstration, requires success (pass / fail) criteria.

TABLE 14-1. Things that can Go Wrong when Designing and Building a Product. A sound process of verification and quality assurance addresses these questions. It also protects against human error, which can affect any of the answers.

Design	Build
– Did we understand or correctly interpret the requirements? – How well did we understand the mission environments? – How accurately did we predict the product's responses to those environments? – How well did we predict the product's critical characteristics? – Did we accurately and completely define the design for Manufacturing? – Did we adequately account for unavoidable variation, such as manufacturing tolerances?	– Did we interpret the design specifications (engineering drawings) correctly? – Did the raw materials and procured parts meet their requirements? – Were the manufacturing processes able to produce products that meet requirements? – How well did we identify and control the process variables that affect the product's key characteristics? – How well did we capture the process in procedures? – How well did the manufacturing technicians understand and adhere to the procedures?

Similarity is another method frequently used, but it is merely a combination of analysis and test: our product is much like another that we have tested, and we use analysis to show the differences are insignificant. Normally this means there are minor design differences, while the mission environments are the same (or less severe), but we can also use similarity for the same design under different environments, as long as sound analysis shows the environmental differences are inconsequential.

Be careful with similarity as a method of verification. We are often tempted to use it without justification. Without dependable analysis addressing the differences, claiming similarity as a method of verification is not sound engineering.

Proactive Versus Reactive Verification

Verification can be proactive or reactive. With reactive verification, we try to find and fix defects after our product is built, and we wait for problems to surface before we address them. Proactive verification means addressing questions such as those in Table 14-1 early in the engineering process. In other words, we anticipate problems and take steps to avoid them. Either approach can work, but obviously we'd rather avoid problems than react to them.

Unfortunately, the typical environment in the space industry rewards programs for minimizing near-term costs, such as those incurred through proactive verification and quality assurance. Winning bids are usually those based on "blue sky" mentalities: problems won't occur (even though we've taken few steps to avoid them). As a result, the typical space program is mostly reactive in its engineering and verification, and the typical space program overruns its budget and misses its initial launch date. If we want to reduce total cost and shorten the time to launch, we must become more proactive in verification.

Of the four basic methods listed above—analysis, process control, inspection, and test—the one we most associate with verification is test. Testing (or inspecting) the end product, though, is a reactive method of verification. In other words, although end-item inspection and testing are effective in detecting defects, they do not improve the product's quality. Our goal is to be more proactive in verification: build confidence with methods that will also build quality into the product. When used this way—as it should be—analysis is a proactive method of verification. We use analysis to steer the design, to build confidence in the design before we commit to it. If we wait until after the design is released to do the analysis, which we now do for the purpose of satisfying a contractual requirement, then the analysis is reactive.[1]

Another proactive method of verification is *development testing*, which is testing we do to acquire information that will improve the design or the manufacturing process. Another way to think of it is that we do development testing to reduce risk. The philosophy here is that early testing of representative hardware is (or should be) a low-cost way to detect and correct deficiencies when we don't have enough information to depend on our analysis. Unfortunately, many large organizations have grown internal bureaucracies that inhibit development testing. What is supposed to be a cost-effective tool for engineers now becomes something that is nearly impossible to get approved.

It's truly refreshing to find an organization at which such things are still simple. I'm presently helping the U.S. Air Force Academy design a small spacecraft for launch on the Shuttle. Because of the ease of building hardware in their own shop—and because of limited analysis capabilities—the team of cadets and staff decided to build and test a development unit for their spacecraft. To keep this unit simple and build it quickly, we decided not to install threaded inserts, instead planning to tap the holes and fasten into bare aluminum alloy. We did not know the pull-out strength for such tapped holes and didn't trust our analysis, so we didn't know how high to torque the bolts. I asked how difficult it would be to tap some holes in a spare piece of aluminum, install bolts, and then torque them until either the threads stripped or the bolts failed. "Not a problem," was the reply. We walked down the hall to the lab and, in two hours, we had the information we needed.

Controlling manufacturing processes is proactive, of course, but to be a method of verification, our controls must be completely effective. Every manufacturing facility I've encountered has claimed their processes are controlled. No one wants to admit otherwise. The important question is, "Controlled relative to what?"

To control manufacturing processes well enough that we don't have to inspect or test the end item, we must have meaningful statistical data demonstrating such control. We can't simply write a procedure specifying the process parameters, such as ranges of temperature and humidity. We must do development testing to demonstrate that those controls keep the products' critical characteristics—such as dimensions, surface finish, or strength—within acceptable ranges or at sufficient levels.

W. Edwards Deming, the man most credited with teaching quality to the Japanese, believed strongly in the importance of process control through statistical methods. One of Deming's arguments was that it's wasteful and costly to churn out products with uncontrolled processes and then screen out defects through inspection. Another was that

[1] It is common, and justifiable, to finish the analysis after the design is released. During design, we try to simplify the proactive analysis with assumptions. Then, once out of the program's critical path, we go back through the analysis with a fine-tooth comb, especially if it is the sole method of verification.

inspection as the means of quality assurance doesn't work. Experience bears this out. The next time you write a document, hand it to an office mate for an inspection, with the objective of finding typos and misspellings. Then enlist a second inspector, and see if the first one found all the errors. Most likely, you'll see what Deming was telling us: Because of human error, inspection alone doesn't work.

Table 14-2 categorizes the methods of verification.

Table 14-2. Proactive and Reactive Methods of Verification and Quality Assurance. In high-volume production, the goal is to rely solely on proactive methods. In the space industry, we typically need reactive methods as well.

Proactive Methods	Reactive Methods
Development testing Analysis – supported by development testing Process control – supported by development testing and analysis	Inspection End-item testing: – Qualification testing (one-time test of a flight-like product) – Acceptance testing (test each flight unit) – Analysis-validation testing (testing to obtain information used to confirm critical analysis)

Despite the desire to eliminate reactive methods of verification, mixing proactive and reactive methods is most cost-effective when production volume is low, such as in the space industry. Learning to control processes completely is costly, and we don't have the production volume needed to justify that cost. Thus, we try to find the best balance of process development and product inspection and test. We control processes to the extent practical and affordable, and then we use inspections and tests to find the few deficiencies our controls are unable to prevent.

Approaches to Product Inspection

Not long ago, space programs routinely inspected products for every feature specified in the engineering drawings. This approach tended to be expensive, so we looked for better ways.

One philosophy that has become popular is *selective inspection*, in which only those features the product team identifies as critical are inspected. This approach is tempting because of the cost savings, but, as with most things, it doesn't come without a price. Selective inspection invites risk in two ways. First, the product team will not know for sure which features will end up being critical. Having control over a product's configuration means always knowing what that configuration is, even when the product is no longer accessible (e.g., in orbit). In 1984, an astronaut in the Manned Maneuvering Unit (MMU) was sent to repair the Solar Max spacecraft in its orbit. Engineers had designed the repair mission based on available documentation regarding the spacecraft's configuration. Unfortunately, the astronaut could not dock as planned because of interference from a small plastic pin in the spacecraft's thermal blanket, which was not shown in the documentation. As a result, the MMU could not be used, and a new method of repair had to be found. With selective inspection, how many of us would have identified the location of that plastic pin as a critical feature?

The second way selective inspection invites risk is more serious: When we identify that a design feature is not critical, the manufacturing people, who are also under the gun to do things faster and cheaper, can interpret this to mean the feature is not necessary or the specified tolerances don't have to be met. Let me give an example.

In the eighth article of this series, I told the story of how I had a hard time becoming confident in the structure for an alignment-critical spacecraft payload. The engineers for the payload developer—our subcontractor—and I had disagreed regarding the extent of analysis needed to verify the structure without a test. At one point, well after the flight structure was built, we decided to have a thorough analysis audit: I, along with another technical representative from my program, met with our subcontractor's engineers at their facility to pour over drawings and analysis with the goal of not leaving until everyone was satisfied. We even brought in a flight-like structural assembly, built at the same time and with the same processes as the flight article and dedicated for other nonstructural tests.

I was most concerned with the welds in the structure, and we spent a great deal of time on them. Looking back and forth between the hardware and the drawing, I noticed something didn't look right. One of the welds specified in the drawing did not appear to be in the actual structure. We called for the lead manufacturing engineer, and he confirmed what we had thought: the weld was not there. How could this be, we asked.

"The welds were not inspected," replied the manufacturing engineer.

"Why not?"

"We only inspected the features the product team decided were critical, and the welds were considered not critical." (Because of high stress margins, as it turned out—which were based on invalid analysis.)

The engineering drawing included over 40 sheets, detailing the parts and specifying the assembly, with thousands of dimensions. Only five of those dimensions were considered critical, and none of those had anything to do with the welds.

This was a "faster-better-cheaper" program; in other words, cost rather than engineering drove it. The welder was also expected to do his job faster and cheaper. Apparently, he found out the welds weren't critical, so, to finish on time, he took shortcuts. Upon closer inspection, we found many of the welds were not of the dimensions specified. (This was a very good welder, by the way.) This is when we all agreed to test the flight structure. You can't very well verify by analysis alone when you don't know which dimensions to base your analysis on!

W. Edwards Deming would have disowned us! He told us to learn to control our processes so well that we don't have to inspect the products, not to quit inspecting our products while also relaxing our process controls!

If you use selective inspection, be very careful not to relax the manufacturing processes. Otherwise, what you think is a noncritical feature might become critical after all. Selective inspection is safest when you are using a process that is well controlled, such as computerized numerically controlled (CNC) machining. Validate the tape (or electronic file) by fully inspecting the first product, then use that tape confidently to machine follow-ons, with inspection of only a few features for a sanity check.

I still believe in 100% inspection when processes are not completely controlled. This approach does not have to be costly. If the dimension is not critical, then open up the manufacturing tolerance. Figure 14.1 shows an example. Not only does this approach reduce the cost of inspection, it more importantly reduces the cost of manufacturing while also decreasing the likelihood of defects. In the example in Fig. 14.1, how hard is it to machine the chamfers to a ±0.1" tolerance? And how hard is it to inspect?

End-Item Testing

As with inspection, testing products before use is warranted in the space industry, even when we've done thorough analysis supported by effective development testing. One reason is the high cost of failure combined with lack of accessibility after launch; another is that experience tells us certain characteristics, such as resistance of electronics to random vibration or shock, cannot be accurately predicted with analysis. A third reason is variation resulting from manufacturing processes that are not completely controlled. Inspection can find some defects, but many can be found only through testing.

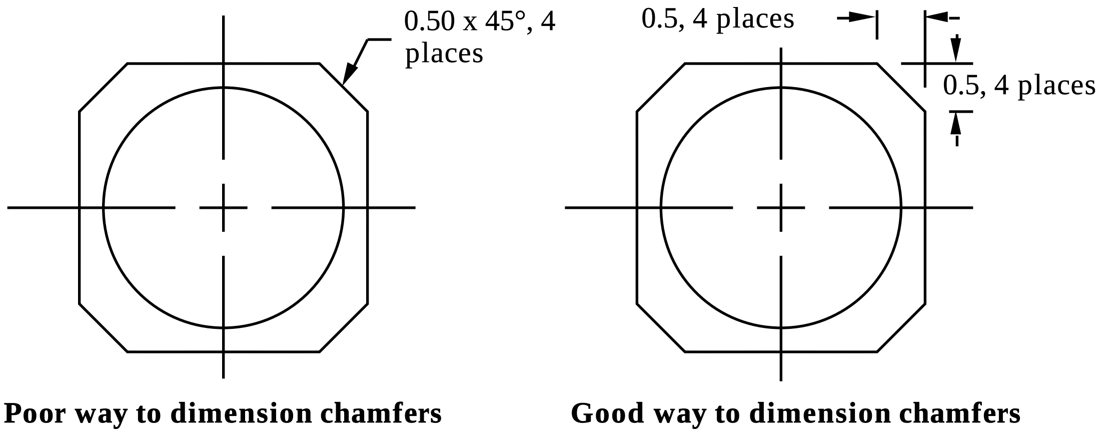
0.50 x 45°, 4 places
0.5, 4 places
0.5, 4 places
Poor way to dimension chamfers
Good way to dimension chamfers

Fig. 14-1. Example of Opening Up the Tolerance for Noncritical Dimensions. This was a shim used in a pipe-flange joint for a spacecraft. All we wanted to do was knock the corners off. When the design was released with the chamfer dimensioned as shown at left, the manufacturing group rejected the drawing and asked for a change. The specified angle had a default tolerance of half a degree, which would have been quite difficult to achieve, given the short dimension. The design was changed as shown at right. The one-decimal-place dimension had a default tolerance of ±0.1", which was much easier to achieve. (Adapted from *Spacecraft Structures and Mechanisms: From Concept to Launch* [Sarafin, ed., Microcosm, publ., 1995])

We do three types of tests at the end-item level:

Qualification test—a test intended to demonstrate the adequacy of a <u>design</u> when analysis alone is not dependable. The *qualification article* is a unit dedicated for test but built to the same recipe (same design and same processes) as for the flight unit. Qualification environmental testing is done under conditions more severe than those we expect the flight unit to ever see in service (*limit environments*), with the difference in environmental level called the *qualification margin*. This margin is intended to provide confidence that the follow-on builds, the flight units, will be able to withstand the limit environments, given the unavoidable build-to-build variation.

Acceptance test—a test intended to demonstrate the adequacy of a <u>product</u> when we are not certain the qualification margin is high enough to account for build-to-build variation. We often do not have meaningful statistical data that shows our manufacturing processes are controlled well enough to keep variation within the bounds of the qualification margin. In such a case, we structurally and environmentally test each flight unit to acceptance levels, which are the same as limit or slightly higher but considerably less than the qualification levels. Most composite structures, particularly those with bonded joints, require acceptance testing, based on typical variation from processes that are hard to control, whereas those made of ductile alloys usually do not.

Analysis-validation test—a test done to acquire information needed to confirm or validate critical analysis upon which verification is dependent. An example is a modal-survey test. Even if we test the vehicle structure to verify strength, the applied loads probably are based on analysis with a finite-element model. The model, though, doesn't perfectly represent the structure—its accuracy is based on the assumptions of the engineer—and thus the loads it predicts are in error. To minimize the error, we do a *modal-survey test*, in which we individually excite the structure's key modes of vibration and measure mode shape, frequency, and damping. We then correlate the finite-element model and repeat the loads analysis. *Thermal-balance testing* is an analogous way to validate the model used for thermal analysis.

Many programs combine qualification and acceptance testing on the first-built flight unit with an approach called *protoflight*. With this approach, to reduce cost, we do not build a dedicated test unit. Instead, we test the first-built flight unit to levels that are usually lower than qualification but higher than acceptance. Depending on the type of construction and on the environment, we then might test each follow-on flight unit to acceptance levels.

The protoflight approach can be risky for the first-built unit. The reason we don't normally fly a qualification unit after testing is that the materials within the unit have incurred fatigue damage that is difficult, if not impossible, to quantify. The same applies to a protoflight unit. Even though the protoflight environments (and duration of exposure) may be less severe than the qualification environments, without doing the qualification tests we have no assurance the design can withstand those environments. For all we know, the protoflight unit may have just barely passed its tests, and little fatigue life might remain for the upcoming mission. To reduce this risk and build confidence in fatigue life, plan to do more extensive development testing and fatigue analysis when using the protoflight approach. Remember: Having sufficient fatigue life is a requirement for all hardware products, whether or not we remembered to specify it.

Although the above discussion is geared toward hardware, qualification, protoflight, and acceptance testing also apply to software. If we develop software a single time and then use it for several missions, with nothing different, we need test the software only once with a protoflight approach. If, for each mission, we must re-enter the code or the input, or if we reassemble software modules, acceptance testing becomes important to protect against human error.

Because of the length of this article, I will stop it here and continue the discussion of verification in Article 15. In particular, I will address the logic of verification planning and ways to responsibly scale back on acceptance testing when production volume increases.

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.