Doing Things Right in Space Programs

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, systems-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 you—and everyone else—have enough time, resources, and freedom to do things right**

Article #19

Keep Everything as Simple as Possible

("… but not simpler!"—attributed to Albert Einstein)

August 2003

Tom Sarafin

President, Instar Engineering and Consulting, Inc.

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

The most effective way to reduce cost through good engineering (Principle #6) is to keep everything as simple as possible (Principle #7). But keeping things simple is important to more than just engineering. And keeping things simple usually also means those things have a better chance of working properly. In this article, I'll address both the technical and nontechnical aspects of this principle.

The FalconGold Example

I've said several times in this series of articles that the key to effective engineering is understanding the problem. Only through understanding can we ensure success when pressured to reduce cost.

Simplicity breeds understanding. We may be drawn by complexity, but a simple system is the only system we will truly understand.

In 1997, a good friend of mine, the late Ron Humble, led a team of cadets and staff at the United States Air Force Academy in a space mission. The mission: measure the strength of GPS transmissions at altitudes above the GPS constellation. The Global Positioning System satellites form a constellation at an altitude of 20,000 km. The satellites transmit to Earth, but the beams are wider than Earth, so a part of the transmission shoots past. Speculation was that there may be enough leak-through transmission strength to enable geosynchronous spacecraft, at an altitude of about 35,800 km, to use GPS for navigation. Before anyone could commit to such a plan, though, someone had to launch a spacecraft and measure the transmission strength. Ron and the Air Force Academy did so—successfully—and confirmed that such signal strength does indeed exist at geosynchronous altitude. As of 2003, several organizations are still using the data that was acquired in that mission.

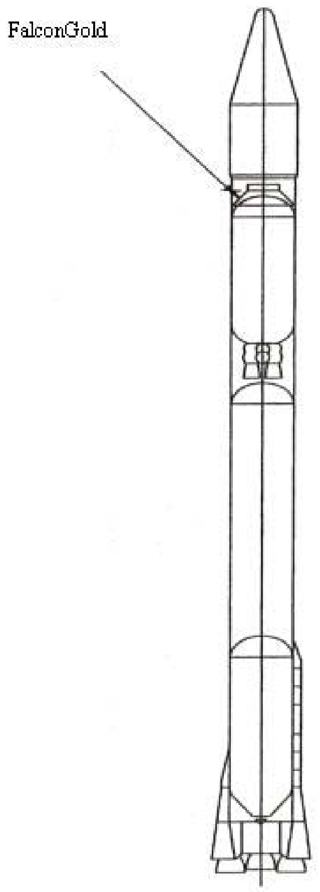
The point of the story is this: The total budget for the FalconGold program was only $150,000, and the mission was designed and completed in12 months.

The phrase "starting with a blank sheet of paper" is overused, but that's exactly what Ron did when suddenly given the budget and an opportunity to fly a secondary payload aboard an Atlas IIA launch vehicle scheduled for launch one year hence with the DSCS III primary payload. The budget was awarded for the purpose of giving a group of cadets the experience of contributing to a space program. The science mission was secondary. It was only after the budget was approved that Ron and his team decided to measure the strength of GPS transmissions at geosynchronous altitude.

$150K for a space mission is unheard of; $150 million is more like it. A typical space program can't pay its accountants for $150 thousand.

Now, to be fair, most of the labor was free for the FalconGold program. Ron and the rest of the staff were paid from other Academy pots of money, and the cadets worked for free. And the launch was provided at no charge. But $150K had to cover all the hardware costs and the cost of testing. (The FalconGold program not only tested their flight unit, they also built and tested an engineering model and a dedicated qualification unit.)

So how did they do it? By keeping things simple. Ron was a very talented, knowledgeable engineer who knew a little (at least) about a lot. His real talent, though, was an ability to simplify. The Atlas would be taking DSCS III to geosynchronous altitude by achieving a geotransfer orbit with its upper stage. Ron recognized that his spacecraft did not need to leave the upper stage. Instead, it could be like a barnacle to that stage, recording transmissions while at geo and sending them down to the ground station when that station was in view while all the time tumbling randomly as part of the upper stage. Thus, FalconGold needed no separation system, no propulsion system, and no attitude determination and control system. Ron recognized that the mission did not need to last long, only long enough to record and transmit a small amount of data, so he did not include a power source other than batteries, which would hold their charge for a couple of weeks. Hence, no solar arrays. The spacecraft included a simple computer, simple software, and a simple communication system, with a low-power antenna. Figure 19-1 shows FalconGold installed on the Atlas.



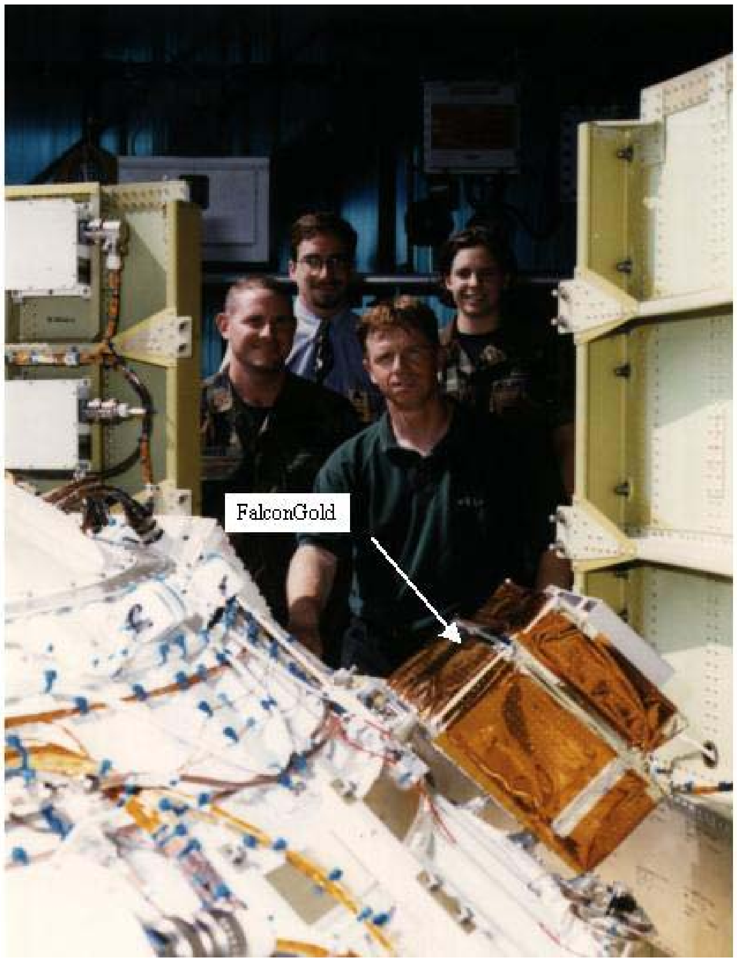


Figure 19-1. The FalconGold Spacecraft, Mounted Ready for Launch. Ron Humble is in the foreground.

Ron negotiated the use of a ground receiver with a 60-foot dish, large enough to enable his low-power transmitter to work. The ground antenna was sitting dormant in nearby Boulder, Colorado. Ron's team designed and built a simple ground station at the Academy to process the data.

Ron made the design so simple that he could keep the entire system in his head. Only through that strategy was the FalconGold system ready for launch 12 months after starting with a blank sheet of paper. The spacecraft transmitted a great deal of useful data before its batteries expired. The mission was a complete success.

Simplifying by Reducing the Number of People Involved

Simplicity of a system can be thought of as being inversely proportional to the number of people on the program, or at least the people making key decisions. In the above example, Ron Humble truly kept his system as simple as possible: he made every key decision. If he had made the system concept any more complex—say, by having the spacecraft separate from the launch vehicle—he would have had to let go of some of the problem to someone else, or perhaps several other people.

Because most of the cost of a space program is in labor, cost is nearly proportional to the number of people involved. Every time we add a human interface, we increase the cost and time it will take to do the job. It takes time to communicate. What's more, each human interface introduces a potential failure mode, the potential for miscommunication.

This is an intriguing premise because it raises the question of cause and effect: Are there so many people involved because the system is complex, or is the system complex because there are so many people involved? I think both are true. Early decisions drive system complexity, which causes us to bring in more people, which in turn adds complexity.

An important strategy for combating this tendency is to prevent people at your organization from becoming too specialized. Encourage and enable them to see larger and larger parts of the problem (Principle #2). Put the few true specialists that you need in staff positions that support programs as needed. This approach will allow each program person to wear more than one hat and will minimize the people involved. Reducing the number of people will reduce cost; reducing the number of people without sacrificing composite knowledge and skill will also increase the likelihood of success.

Of course, the opposite is also true: Staffing the program without the requisite skills may initially reduce cost, but that will eventually change as the program responds to a multitude of costly problems. We have to invest in Principle #2 before we can effectively reduce staff size.

Subcontracting may appear on the surface to reduce cost, but it often increases cost because it adds people and interfaces. In the long run, it may be better to develop in-house skills or to keep paying the overhead associated with maintaining your machine shop.

Politics can drive us to add human interfaces. As an example, consider government research centers. If your center raises money for a project in an area of research in which, say, three other centers are involved, each of those centers may want a piece of the pie. Involving these centers may keep you popular, but it will certainly drive cost. All such interfaces do.

Simplifying Requirements

We want our set of requirements to fully capture what the system must do and all the constraints under which it must do it. Beyond those two objectives, requirements add unnecessary complexity and thus drive unnecessary cost. As addressed in Article #8, requirements related to quality or probability of success also take responsibility away from the people who should have it. Finally, a massive collection of requirements will go unread by the product team as they attempt to meet their responsibilities. People can keep only so many things in their heads; they will simplify their job out of necessity, even if someone above them tries to complicate it. And buried within that collection of requirements will be a key one that goes unrecognized.

Rather than setting out to identify all the things that can possibly go wrong and then writing a requirement for each intended to stop it from happening, focus on identifying the true requirements: what the product must do, how well it must do those things, and under what constraints must it do them. The set of unique requirements should be as concise as possible and clearly set apart from controls, criteria, and plans for verification, all of which should be defined outside the product specifications and standardized as much as possible between programs so that the engineers don't have to think very hard to remember them.

Make the phrasing of requirements clear, so that there is only one interpretation for each. This will take a lot of work—an investment. Several waves of review and editing will be necessary, a process similar to that in writing a good book. Nothing else on the program, though, will pay as great of a return from such an investment.

"Our life is frittered away by detail. Simplify, simplify!"—Henry David Thoreau

Simplifying during Conceptual Design

We engineers, by our nature, want to dive into details and solve the entire problem at once. A sign of maturity as an engineer is the ability to overcome this tendency. Consider the example of conceptual design of a space or launch vehicle, an activity requiring the contributions of many disciplines and, hopefully, a systems engineer who can see the whole picture. The person configuring the vehicle should be such an engineer. If he or she takes time to detail the structural design, for example, it may take a month or more to finish a configuration drawing. Meanwhile, the rest of the design team is off working the details of their own subsystems, such as attitude determination and control, based on assumptions related to the configuration that are not mature enough to warrant such detail (e.g., the distance between the spacecraft's center of pressure and its center of mass, which affects the disturbance torque).

Spacecraft design is iterative. For the physical aspects, what the team needs during conceptual design is someone who can pull together a rough configuration in a few hours and then send it out for review—even though it's far from perfect—and then draw up alternate configurations for comparison.

If you're having a hard time grasping the importance of a configuration drawing, pay close attention the next time you're involved in conceptual design. See what happens to a meeting when someone shows the configuration for the first time. Everyone starts to see problems with the concept for their subsystem that they had been so confident in based solely on parametric analysis. Inadequate fields of view for antennas and sensors, solar panels that won't see enough of the sun, parts of the vehicle that will cast shadows on solar cells, heat-generating components without adequate radiating surfaces—these and many other problems for a spacecraft design are identified only after a configuration drawing is distributed. Such a drawing can't be withheld by the person creating it in order to complete details, such as bolt patterns and machined ribs, that have no bearing on the problem at hand: finding a configuration that works for everyone. The configuration designer must be a master of the art of simplification.

A complex system has more potential ways it can fail than does a simple system. The decisions that drive system complexity are made mostly during conceptual design. The following are examples of complexity in spacecraft design:

- Three-axis control: If this is really necessary, the cost of your program may double, and failure will become much more likely. Can you meet mission objectives by controlling one axis while using a gravity-gradient boom and magnetic torquers to control the other two? Better yet, can you get by without any control, allowing your spacecraft to tumble randomly?

- Active thermal control: If we define *active* thermal control as requiring power and closed-loop control, such as heaters or louvers controlled based on measured temperatures, such thermal control not only is costly but also adds to our spacecraft's power demands. Adding power means there will be more heat to reject. Keeping the thermal control system as passive (simple) as possible requires intelligent location of spacecraft components, making use of environmental heating for non-powered devices, shaded radiating surfaces for heat-generating items, and spreading heat loads throughout the vehicle.

- Deployable appendages: Scaling back on performance so you can get by with body-mounted solar cells rather than deployable arrays will save not only the cost of developing mechanisms but also of testing them. It also reduces risk.

Simplicity should be foremost in your mind when establishing the operations concept for your spacecraft. The point in operations in which you want the spacecraft to acquire certain data until the time you receive it will encompass many interfaces between people, hardware, and software. The same is true for operating a launch system. The goal is to minimize the people involved in operations (recurring cost) with software, yet to keep the software as simple as possible to reduce development cost and potential failure modes. If not considered thoroughly during conceptual development of your spacecraft, software will become far too complex, will end up driving the program schedule, and will be impossible to test thoroughly given all the possible situations. The best software is designed in modules, with simple interfaces, just like hardware, so the modules can be tested separately before testing the system, and so the system can be tested at all.

Simplifying by Standardizing

Most engineers resist standards because they believe standards overconstrain them and stifle creativity. This is indeed the case when standards are overused. Effective use of standards, though, reduces cost by minimizing design options and things that can go wrong.

Interfaces, attachment hardware, and electrical connectors are great candidates for standardization. As an example, consider fasteners. Without standards, each design engineer will select fasteners with which he or she is familiar—driving the need for many different tools for installation—or, if inexperienced, will select them from scratch. Standardizing fasteners allows you to minimize fastener types and thus helps you avoid problems. It also makes it feasible to stock spares. “Just in Time” may be an effective management approach for some operations, but not for one-of-a-kind spacecraft using fasteners that have a three-month lead time. Bolts do cross-thread, grip lengths are often improperly selected, and self-locking nuts do stop locking and need to be replaced after being installed and removed several times.

Another good candidate for standardization is materials. How silly do we look when our spacecraft suffers a short circuit or a mechanism jams because of whisker growth on the cadmium plating on a bolt or nut? Long ago the space industry recognized that cadmium (and zinc and tin) grows whiskers in a vacuum; without standards (and controls), though, someone will make this mistake again. How do I know this? Because it's happened more than once. Without standards, someone will select fastener materials that corrode when mated or that gall and cannot be disassembled without a drill. These mistakes cost money, and they happen over and over again.

Standardization is for the no-brainer stuff, the stuff we've already figured out. Standardizing interfaces, fasteners, materials, electrical connectors, and other such things not only leaves less that can go wrong, it also frees up the engineers so they can devote their energy to the unique problems of the program. Can you imagine how expensive your house would be if every part, component, and connection were designed from scratch? Yes, the space industry has low-volume production, but we can standardize many elements of our vehicle and take advantage of volume discounts.

Sometimes the principle of keeping things simple conflicts with the desire to instill ownership and responsibility (Principle #3). We have to find the right balance. Many, myself included, like to criticize the mountain of NASA standards and requirements applicable to folks who want to fly a payload on the Shuttle. But quite a few of those standards actually keep things simple for the payload organization and thus can reduce cost. For example, consider the problem of potential contamination of other payloads from material outgassing in space. NASA's Hitchhiker program for small secondary payloads, as an example, provides a list of acceptable materials. (Note that this list is also intended to ensure materials have adequate resistance to stress-corrosion cracking.) If your small payload contains any materials not on the list, you'll need to convince many people those materials have acceptable outgassing properties (and corrosion resistance) in order to attain a waiver. This may sound like over-constraint by NASA, but, if you understand the requirement up front, it's usually not difficult to design a payload only with the materials on the list. In fact, knowing the consequences is what gives a program incentive to meet the standard.

Consider for a moment how you might handle this situation if you were responsible for integrating multiple payloads with a launch vehicle. You try not to over-constrain the payload folks, so let's say you don't issue an acceptable list of materials, and you levy no requirements pertaining to outgassing of materials. Instead, you schedule bi-weekly telecons with representatives from each payload organization to address contamination concerns. Each payload group is initially free to use whatever materials they want. However, as the telecons progress, payload organizations gradually find out that some of their materials are not acceptable to the other payload groups, so they have to change their design—or take costly measures to prevent contamination.

Which approach would sound better to you if you were planning to design and launch a small satellite? Standards, when used wisely, reduce cost as well as potential problems.

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.