Acquisition Archetypes

Changing Counterproductive Behaviors in Real Acquisitions

Everything for Everybody

Measure Twice, Build Once

A program sponsored by several services was trying to build a software infrastructure for communications that could be used on platforms for air, sea, and ground vehicles. A common system offers significant cost savings over custom software for each different platform.

Before the contract award, five platform programs agreed to use and fund the infrastructure software, placing it on the critical path of their schedules. It was important for the program to have platforms committed to using the software and contributing to fund the development—that demonstrated need and helped defray costs. That commitment by the initial five platform programs generated the interest of still more programs—and necessitated discussions about new infrastructure requirements needed to support these additional platforms.

Too Much of a Good Thing?

Commonality is a great objective, but sometimes there can be too much of a good thing. In this program, it meant that the infrastructure software was going to have to be deal with multiple platforms with varying requirements—a capability that would come at a steep price in terms of cost, effort, and complexity. As one engineer on the program observed, the system "has to be complex to do this job. It could only be simpler if the requirements were fewer, or simpler."

The program "needed platforms to get funding, but that means taking on differing requirements," said one program lead. "So it has to be configurable, which brings in software complexity." Complexity translated to additional development time and effort.

The number of platforms that needed to use the software to sufficiently amortize the cost meant that more custom requirements had to be addressed and resolved. "We wanted to involve [the platform programs] as early as possible, so they could cooperate," one manager noted. "That keeps them involved, but it allows them to drive design, and push us off track."

Running Out of Time

In order to keep the platform programs committed to using the infrastructure software, the team had to rush to meet the "need dates" for the various platforms. This forced the program into an aggressive, 18-month schedule—a schedule in which everything would have to go like clockwork. It was clear, said one team member later, that the infrastructure program "was trying to do too much in too little time." A key program management review with the contractor held the next year showed that they were falling behind.

Jumping Ship

The program lost more momentum as tight budgets and funding cuts forced two intended platform programs to drop their plans for the new infrastructure. Not long after that, a third, key platform program decided the cost growth was too much, and with it, one of the participating services then backed out of the infrastructure program entirely, and all of its platform programs went with it.

Cost wasn't the only issue, according to one manager. "In most cases platform programs have pulled out due to schedule slips," he said. "[The infrastructure program] can't deliver the capability required in the timeframe the program has to have it."

As the number of platforms declined, new requests still came in to the infrastructure contractor from the remaining platforms, whose happiness had now become a critical concern. As one frustrated team lead put it, "Every time [the platform customer] said we must do this to make it work, we rolled over and agreed to do it."

> *"They tried to be too many things to too many people…"*

You Can't Please Everyone

The program continued on, unable to amortize its costs across just the remaining platforms, a year behind its original 18-month schedule, and unable to justify the additional development needed to support more platforms. "This would have been a totally different program if we didn't need to build a general-purpose solution," observed one member of the program staff.

In the words of a program official, "[they] bit off more than they could chew, trying to do everything for everybody. You sacrifice too much, making it too complex. If you had scaled it down a bit it could have been done faster, and more easily."

(Continued on page 2)

(Continued from page 1)

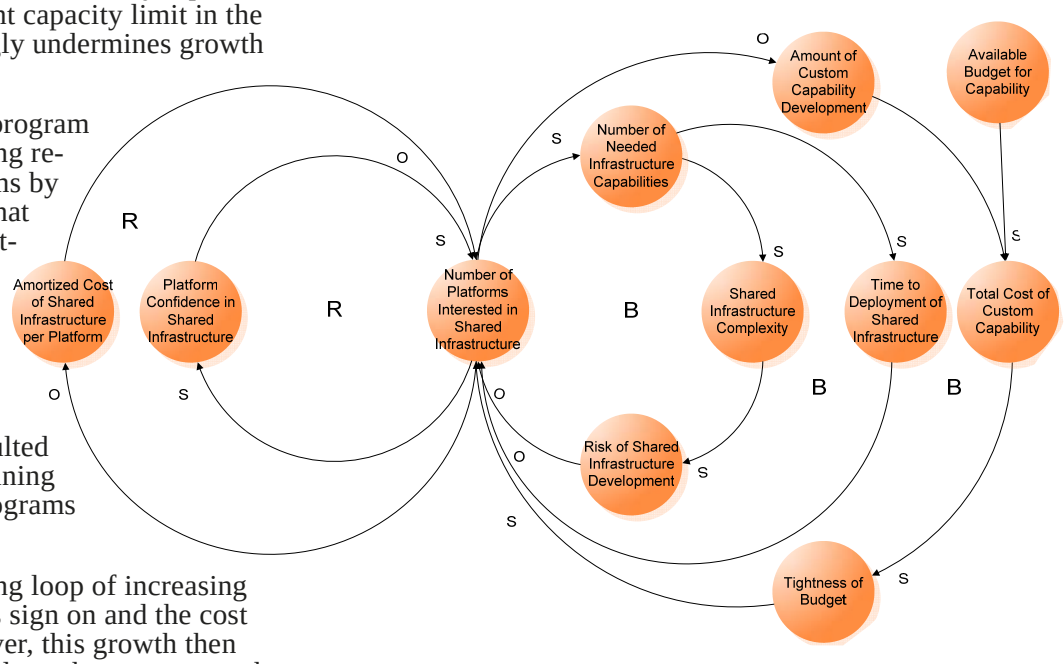
The Bigger Picture

This acquisition archetype is an instance of the system archetype "Limits to Growth," in which initially rapid growth slows because of an inherent capacity limit in the system that worsens and increasingly undermines growth as more growth occurs.

In our example, the infrastructure program attempted to reconcile the competing requirements of the different platforms by creating a single software system that would fulfill all of the different platform requirements. Furthermore, the platform programs made their aggressive delivery schedules a requirement for success. When the infrastructure program slipped and cost grew, the platform programs opted out. This loss of funding resulted in further cost growth for the remaining platforms, in turn causing more programs to back out of the program.

This archetype presents a reinforcing loop of increasing platform interest as more platforms sign on and the cost per platform steadily drops. However, this growth then potentially reverses by a balancing loop that represents the side effects of increasing complexity, cost, and delivery schedule from the number of participating platforms, that undermines and erodes platform interest.

A Causal Loop Diagram of "Everything for Everybody"



System variables (nodes) affect one another (shown by arrows): Same (S) means variables move in the same direction; opposite (O) means the variables move in opposite directions. Balancing (B) loops converge on a stable value; Reinforcing (R) loops are always increasing or always decreasing. Delay denotes actual time delays.

Breaking The Pattern

Once this dynamic starts, it is difficult to stop. Prevention is clearly the best remedy.

Platforms that have schedules too short for the infrastructure program to realistically meet should not be considered as viable candidate participants. Furthermore, the program office should act to "hold the line"—to avoid letting the attraction of more funding and support force unwise decisions regarding the number of capabilities that can be delivered by a single system. This can be done by evolving the set of infrastructure capabilities slowly and modestly based on return on investment—starting with the smallest set of capabilities that will provide the highest value to a small set of platforms.

One other option that can minimize this dynamic is to either provide incentives for (or even mandate) the use of the common infrastructure by individual platform programs. Mandates, however, may be unpopular because they impose an external dependency on a program over which the program office has no control—and can thus become a risk. Incentives can take the form of economic advantages offered to the platform program to balance the additional potential risk the program takes on by choosing to use the infrastructure.

Short of an incentive or mandate, potential platform programs could be required to do a cost-benefit analysis between using the infrastructure solution and developing a custom solution. This approach at least ensures that the program is aware of the cost savings that are possible by using the infrastructure, and that the risks of both approaches are weighed.

After the infrastructure has been successfully delivered and integrated at least once, many of these issues become moot. Credible numbers for integration cost and effort should now exist, as well as data on performance of the system in the field—both addressing many of the risks that otherwise might affect potential platforms.

Software Engineering Institute
Carnegie Mellon

Acquisition Archetypes is an exploration of patterns of failure in software acquisition using systems thinking concepts. It is published by the Acquisition Support Program of the Software Engineering Institute.

For more information, visit http://www.sei.cmu.edu/programs/acquisition-support/