The FIST Approach To System Development Projects

LtCol D. B. Ward, USAF

Abstract - This paper describes the impact of a value set called FIST (Fast, Inexpensive, Simple, Tiny) on system development projects. The findings show the FIST value set enhances project stability, increases the project leader's control and accountability, optimizes failure, and facilitates learning. FIST is therefore recommended as a productive set of values for system development projects.

I. INTRODUCTION

Military weapon systems in general, and C4ISR systems in particular, tend to be very complex and to experience significant overruns and delays during their development. However, the complexity, costs and delays often associated with such projects are not inevitable. It is possible to develop and field high-tech military systems on short timelines and with small budgets, if appropriate restraint is exercised in several key areas.

The root of the nearly ubiquitous problems with defense acquisition is not fundamentally technical, nor is it to be found in processes and procedures or in oversight or lack of oversight. A historical review of acquisition successes and failures reveals that in every decade the exact same regulatory environment produced some excellent programs that delivered on time / on budget and some failed programs that experienced huge problems and delivered nothing at all. This paper proposes that the key difference between successes and failures is the underlying *values* that drove the project leaders' decision making. [*Note: the term "project leader" includes program managers, engineers, customers and others who make decisions which impact the project*].

Values are statements of priorities and preferences. An organization's values typically answer the questions "What is important?" or "What is good?" The answers to these questions shape the measures of merit project leaders use to assess and guide a system development project. Values are in essence meta-requirements, which decision-makers use to assess the validity and worth of other requirements within the system, function, organization or process.

Improvement efforts which focus on changing external attributes like technology, oversight, process, organization and training, risk overlooking the way values impact decision making and organizational behavior. The FIST approach, however, is built on the observation that the project leaders' values are a determining factor of the project's outcome. The challenge is to identify and establish a productive set of values to apply to the design of organizations, processes and technologies.

II. THE FIST VALUE SET

A value set is a collection of values that shape decision-making. System development projects can be guided by (and judged by) a variety of value sets, each of which contains different preferences and priorities. These value sets may or may not be conscious and explicit, and may contain internally contradictory elements.

Military projects often conform to a value set that includes a preference for complexity, largeness, comprehensive rationalistic assessments, perfectionism and risk aversion. Thus, the size and complexity inherent in the system and the organization are seen as signs of sophistication. In the C4ISR realm, they are often seen as inevitable attributes, as if a good C4ISR system cannot be simple, inexpensive or developed quickly. Of course, the Agile approach offers an interesting counterpoint to this perspective, as does FIST.

Project leaders using this "traditional" value set generally view budgetary and schedule constraints as obstacles to be overcome. They may make statements such as "It is important that we take our time and do this right," or "If we just had more money and more time, we'd be able to fix these problems." Project leaders on both the government and contractor sides are also strongly incentivized to maintain high expenditure and obligation rates, in order to prevent cuts to future budgets.

The traditional value set is often accompanied by a conflicting desire to deliver a new system on time and on budget. The divergent values creates friction as decision makers seek to keep costs low while simultaneously keeping spending rates high. This requires project leaders to perform an unfortunate type of mental gymnastics, in which they represent increases to the budget and schedule as something other than an increase in time or money. Terms like "rebaselining" are used to resolve the mental conflict caused by the perceived need to take action contrary to the desire for keeping costs low and schedules short.

In contrast, the FIST value set encourages project leaders to minimize the budget, schedule, team size and complexity of their effort, across the entire development. Project leaders who subscribe to the FIST values view budgetary and schedule constraints positively and might explain "Constraints foster creativity," and make statements such as "It's important to deliver this capability quickly." Using the FIST values, project leaders generally build small teams with austere budgets and short delivery timelines. FIST drives the creation of simple organizational structures and tightly-focused systems, which are generally built on mature technologies, using modular architectures and open standards.

These behaviors and design approaches specifically target common problems encountered in C4ISR projects. For example, in addition to being simpler, modular, open designs help ensure the new system can be integrated into an existing, architecture and responsive to future changes. The FIST approach explicitly asserts that a well-executed capability which cannot be integrated into the operational environment is in fact *not* well executed, because the additional integration effort adds time, cost and complexity, reducing its reliability and utility.

The following sections describe how the four components of FIST can be expressed within a system development organization, along with the ways each piece contributes to and supports the other pieces.

A. The Fast Value

The F in FIST stands for Fast. Organizations often express this value by taking steps to encourage and reward short development timelines. In an environment where Fast is valued, participants are rewarded for meeting or beating contract deadlines, milestones and delivery dates. When Fast is not valued, schedules tend to be long, and requests for schedule slips are viewed as a reasonable approach to correcting difficulties.

The Fast value is not consistent with the superficial appearance of rapid progress. Merely giving in to short-term, tactical time pressure and clumsily "speeding" tends to result in the system development effort taking *more* time, not less, as project leaders inappropriately cut corners on things like early planning or developmental testing. This ultimately results in a slower delivery, often because it delays discovery of significant problems which end up needing to be corrected later, at much greater expense and time. Being hasty—i.e. speeding—is therefore not consistent with the assertion that "it is important and good to be fast," because hastiness delays the system's delivery. The Fast value, in contrast, takes a strategic point of view, and is not satisfied with the shallow appearance of speed.

The Fast value leads program managers to trim unnecessary activities. The danger is that it also tempts them to cut essentials. Project leaders must thoughtfully assess how to express the Fast value effectively, without injecting undue risk to the overall outcome. The best approach appears to involve relying on experienced, talented program managers who are wise enough to distinguish between necessary and unnecessary activities when making decisions related to speed.

B. The Inexpensive Value

The I in FIST stands for Inexpensive. Organizations express this value when they deliberately pursue low-cost solutions and establish contractual incentives to reward cost under runs. They make formal commitments to maintain their budget and are willing to sacrifice other attributes, including performance or redundancy, to ensure the program cost stays low. When being Inexpensive is not valued, project leaders seek to increase their budget as a means of solving problems, and work hard to ensure that next year's budget is not decreased.

The Inexpensive concept is based largely on research by Pierre Sprey [3] and Fitzsimonds & van Tol [4]. In a widely cited research project, Sprey argued for the superiority of "cheap winners" over "expensive losers." Similarly, Fitzsimonds and van Tol's research indicated that "militaries are driven to innovate… by the need to make more efficient use of shrinking resources…" They conclude "Innovation is not necessarily or even primarily a function of budget." [4]. This inverse relationship between cost and effectiveness is one of the reasons often given for holding to the Inexpensive value.

The Inexpensive value is closely related to both Fast and Simple, as speed and simplicity are often pursued for the sake of keeping the overall costs down. If a project has a tight budget, the schedule must also be short enough to deliver the system before the money runs out. Similarly, if a project has a short schedule, there simply is not enough time to spend a lot of money.

C. The Simple Value

The S in FIST stands for Simple. Organizations often express this value by reducing complexity in their organization and procedures as well as in the technology systems they develop. Program managers who value simplicity tend to focus their projects on a narrow, stable set of modest operational requirements and aim for "the shrewd application of available technology," [1], rather than a frequently changing and wide-ranging collection of unproven, high-tech, beyond-the-state-of-the-art objectives. When Simple is not valued, project leaders tend to view complexity as a sign of sophistication and pursue complex, high-technology solutions for their own sake, rather than because they meet the actual operational needs.

The FIST value set draws a clear distinction between simple solutions and simplistic solutions, a point discussed further in *The Simplicity Cycle* [10]. In this context, the Simple value is built on a foundation of experience and complexity. Using mature, proven technologies as components (i.e. TRL 7+) is therefore an example of mature simplicity, building as it does on previous experience. The converse is also true—relying on new developments and unproven solutions leads to solutions which are either inappropriately simplistic or unnecessarily complicated.

Because integration is such an important capability for virtually every 21st century technology system, and particularly so for C4ISR systems, a project leader's appreciation for the systems engineering principle of high cohesion / low coupling can have a major impact on a system's complexity. This principle basically states that each of a system's components should be designed such that it provides a small number of capabilities (High Cohesion) and such that changes to one component do not drive substantive change to the rest of the design (Low Coupling). High cohesion and low coupling greatly simplify a system's development, as well as related testing, operations and maintenance efforts.

Along with the aforementioned high cohesion / low coupling principle, FIST points to a number of other specific tools and techniques for technically simplifying projects. These include mature, well-documented tools like the TRIZ matrix of 40 principles. The Modular Open System Architecture (MOSA) approach, which was integrated into DoD 5000.1 in 2004, provides a powerful set of guidelines that foster simplicity, both organizational and technical. Similarly, the Agile literature and, less formally, documents like the online *Zen of Python*, all offer roadmaps for good design practices that are based on an appreciation for thoughtful simplicity.

Organizationally, simplicity means relying on trust, talent and experience to achieve objectives, instead of elaborate, formal (i.e. complex) procedures. A simplified organization avoids erecting intricate mechanisms to audit, monitor, control

D. The Tiny Value

Finally, the T in FIST stands for Tiny. Unlike the previous three values, Tiny is seldom invoked explicitly in hardware development projects, at least not using that particular term. However, Tiny is often practiced and pursued under the labels like lean, streamlined, trim, agile or efficient. Organizations express this value by fielding small teams, using miniaturized components, building small systems or requiring minimal documentation, among other behaviors. This may be expressed in the phrase "as much as necessary, as little as possible." When Tiny is not valued, project leaders assemble large teams and large systems, judging the size as a measure of merit.

Tiny can be understood as the cornerstone concept behind the FIST value set, but it is also perhaps the most difficult to put into practice and the most rare to find. This is particularly true in environments like traditional NASA or DoD system development communities, which tend to reward and encourage largeness. However, given the prominence of Lean development approaches, Agile Acquisitions and Extreme Programming methods, the Tiny value is becoming more popular and common, under a variety of labels.

III. BENEFITS OF FIST

While the FIST approach does not ensure a positive outcome on every development project, the FIST approach conveys several significant benefits. Specifically, the FIST approach enhances project stability, increases the project leader's control and accountability, optimizes failure and facilitates learning.

A. FIST Enhances Project Stability

Project leaders wage a constant battle against instability and uncertainty, whether caused by changes in technology, changes in the environment or changes in the political/financial situation. These changes are often unforeseen and unforseeable, and as Taleb points out, "the unexpected almost always pushes in a single direction: higher costs and a longer time to completion." [9]. Pursuing the Fast, Inexpensive, Simple and Tiny values, provides a way for project leaders to inject stability across several fronts.

1) Fast Stability

Program instability comes from many different sources, but the primary origin for all these sources is simply *time*. Given enough time, new discoveries and breakthroughs will render previous technologies obsolete. In a military context, old enemies are defeated while new enemies emerge. Political leaders come and go, as do program managers, project leaders and system architects. Economies expand and contract. Each of these changes can cause changes to a project's structure, objective, funding, design, priority and schedule. Our inability to accurately predict these changes contributes to the cost and schedule overruns which are so prevalent in DoD and NASA projects. Further complicating matters is the high "impact of forecast degradation over long time periods."[9]. Left uncorrected, a small forecasting error's impact grows over time, so by emphasizing the importance of short development timelines, we seek to minimize forecasting errors.

In contrast to traditional development approaches, the FIST approach reduces a project's exposure to the unexpected and minimizes this particular source of instability. Project leaders who embrace the FIST values insist on short timelines and make decisions that support rapid delivery of the required capabilities. By valuing speed, they aim to deliver capabilities before too many changes are manifest. This does not mean all change will be avoided, but both the quantity and significance of the change will be greatly diminished.

2) Inexpensive Stability

While a short timeline provides a great deal of a FIST project's stability, a small *budget* also conveys stability. This is largely because FIST projects are not very tempting sources of funds when budget cuts come down the line, for the simple reason that a FIST project's budget is already small. It has essentially been pre-cut, and further decreases to a Tiny budget are likely to be seen as not only unfair but also unproductive and unlikely to make a dent in the organization's overall finances. It simply makes more sense for budget cutters to go after deeper pockets and projects that are better able to survive a reduction.

The combination of a short timeline and small budget means the project is more likely to be fully funded from the start, rather than having to deal with the uncertainty of requiring budget authorities to authorize future (large) budgets. The near-term delivery schedule also means budget cutters are less likely to cut a current-year budget and promise to "repay" it with future dollars, because the project is likely scheduled to deliver before the funds can be repaid

Finally, new regulations tend to focus on high-profile, big-ticket projects. FIST-driven projects with sufficiently small budgets can remain appropriately below the radar and thus minimize exposure to the unexpected changes that often accompany new regulations.

3) Simple Stability

Simplicity fosters stability, both technically and organizationally. From a technical perspective, the Simple value's emphasis on mature technology tends to reduce the uncertainty and instability inherent in cutting-edge, not-quite-proven technologies. The principle of high cohesion / low coupling, which is also a key technical element of simplicity, means that the impact of change is localized, with limited ripple effect through the rest of the system.

Of particular relevance for C4ISR projects is the so-called State Explosion Problem. For this context, the problem is defined as follows: For any system, the number of possible states increases exponentially with the number of dynamic components. As the number of possible states increases, the effort, cost and time associated with testing the system also increases and its operational predictability decreases. Excessive complexity can therefore render a system essentially untestable and unpredictable, increasing the likelihood of unexpected and undesirable performance when entering unanticipated, untested states.

Simplifying the system's architecture mitigates or even avoids the state explosion problem, without resorting to strict performance and usage restrictions. Thus, the benefit of simplifying a C4ISR architecture extends beyond the cost and time required to develop the system. Simplicity actually conveys greater freedom to the end user because the system's performance is more predictable.

Further, a simpler, more streamlined organization tends to have faster and clearer communication than a large, complex organization. This communication clarity reduces the amount of instability caused by miscommunication and bureaucratic inefficiency, and incidentally increases accountability, as discussed in the following section.

The bottom line is that the FIST values help projects present a smaller target to the forces of change, whether those forces are financial, technical, political, legislative or some other category. This does not mean FIST makes projects more *resistant* to change – rather, they are simply confronted with fewer instances where change is necessary.

B. FIST Increases Control And Accountability

Technology systems development efforts typically include a wide range of stakeholders. For the DoD and NASA, these stakeholders include many layers of bureaucracy and several branches of government. Involvement by such an extensive community often limits the amount of influence a project leader can have on the project he or she is responsible for. This is unfortunate because the project leader arguably has the most accurate, relevant and timely information, particularly when compared with individuals who are several layers removed from the project or who are from other organizations entirely. By enhancing and localizing the project leader's *control*, FIST also increases the project leader's *accountability* for the outcome.

Writing in *Defense Acquisition Review Quarterly*, Christensen, Searle and Vickery examine the question of programmatic influence, drawing an interesting distinction between cost growth and cost overruns on acquisition programs. They define cost growth as "the difference between the *initial* budget and the final cost of the program," while cost overruns are defined as "the difference between a contract's *final* budget and final cost." [2, emphasis added].

The authors observe that the traditional definition of cost growth fails to make a distinction between factors which a program manager can influence and those factors which are beyond a project leader's scope of influence (i.e. changes in technology, changes in the threat environment, congressional decisions, etc). They suggest project leaders have little influence on cost *growth*, and thus can not be held accountable for it. Cost *overruns*, on the other hand, derive from factors which project leaders can influence. So, they recommend cost *overruns* as a more appropriate metric for determining programmatic efficiency.

It is indeed important to distinguish between factors a project leader can influence and those that are outside their control, so there is much merit to this argument. However, project leaders actually have more influence over the so-called external factors than Christensen et al seem to think. It is not necessary to simply accept cost growth in an acquisition project as an unavoidable fact of life, *unless we also accept long timelines as inevitable and unavoidable.* As the previous section explained, by limiting the amount of time spent on a project, FIST gives project leaders the opportunity to minimize many of the destabilizing external forces that lead to cost growth.

The introduction of change from the external environment is beyond the control of the project leader. However, by using a small budget, a short schedule and a streamlined team, FIST minimizes the program's *exposure* to these externally-driven changes, retaining control, influence and accountability at the local project leader's level instead of allowing external actors to take control of the project. In fact, the project leader's influence is inversely proportional to the size of the budget and schedule. Nowhere is this seen more starkly than when a project needs to be cancelled.

The simple reality is that big, complex and expensive government projects tend to create a lot of jobs, in multiple congressional districts. This is considered good for employment, but it means various government Agencies and Departments are sometimes forced to continue projects they would prefer to cancel. Elevating this kind of decision making ability to the Congressional level effectively reduces the project leader's influence and accountability. The DoD could avoid this problem by not permitting these conditions in the first place.

Congressmen and Senators are appropriately concerned with protecting jobs in their districts and representing the interests of their constituencies. This is what they are elected to do, and such protective actions are entirely consistent with their legitimate responsibilities. However, when a government agency or department launches a big project and spreads the work across dozens of states, large numbers of jobs across the country are all tied to a single system. Cancelling such a system causes pain in many locations, and Congress naturally tries to prevent such a negative economic impact. At that point, Agency and Department heads often find it difficult or even impossible to cancel the project, even if it is having significant problems.

In contrast, FIST projects are deliberately too small to spread across a wide range of Congressional districts. By design, FIST projects are developed by Tiny, co-located organizations (which may be part of a larger entity, as with Lockheed Martin's Skunk Works division). When a FIST project begins to go south, it can be cancelled without any given district losing very many jobs, relatively speaking, and before much time has passed. Similarly, the amount of money being spent on any given FIST project is relatively low and is therefore easier to write-off as a research investment, leading to less hand-wringing over waste than the cancellation of a $20 billion project.

C. FIST Optimizes Failure

Neither the traditional approach to system development nor the FIST approach guarantees a successful outcome. Given enough attempts, a certain amount of failure is inevitable, regardless of approach. But while we cannot avoid failure entirely, we can influence the kinds of failure we experience. Nicholas Taleb suggests striving to create "situations where favorable consequences are much larger than unfavorable ones…" [9]. That is, we ought to pursue situations where the benefits of a positive outcome significantly outweigh the cost of a negative outcome. One of the ways to do this is to minimize exposure to loss. The other is to ensure that any negative outcomes become learning experiences and building blocks for future endeavors.

The ideal failure, therefore, is one in which little is lost and much is learned. Such a failure could be termed an *optimal failure*. An *epic failure*, in contrast, is one in which much is lost and little is learned. TABLE 1 illustrates these two failure types.

When FIST projects fail, they fail optimally. These failures are realized before much time and money is expended, so losses are small. They also have a high probability of conveying meaningful lessons, because on a short schedule, project leaders are more easily able to witness the impacts of their decisions and learn from their experience.

Unfortunately, epic failures are arguably the only kind of failure a large project can experience. When a big project fails, it fails spectacularly, costing a lot and teaching too little, too late.

Whenever large quantities of time and money are expended before the outcome is observable, the project is exposed to significant loss. If a large percentage of participants have moved on to other projects and/or retired before the failure is observed, the opportunity to learn is low. Even if the original decision makers are still around and directly witness the consequences of their actions, it is often too late to apply the lessons very much further. This is because learning requires both *observation* of the phenomena and *timely reflection followed by action*, neither of which are likely in big, lengthy, expensive projects.

The FIST approach requires a significantly different perspective on failure than the standard methodology. In the traditional approach, it makes sense to measure failure rates on a per-attempt basis (i.e. failures per-cohort or per-portfolio), because each attempt is expensive and takes a long time to realize. But when attempts are quick and inexpensive, as in the FIST approach, a relatively high failure rate is more acceptable, or perhaps even irrelevant. Indeed, a relatively high per-attempt failure rate should perhaps even be demanded.

D. FIST Enhances Learning

Failure is not the only way to learn, although it is one of the most fruitful. A diverse set of successful experiences can also help produce agile-minded, well-rounded and creative project leaders. Unfortunately, the traditional development approach, particularly as currently practiced in the field of manned aircraft development, is trending in the opposite direction.

A 2005 presentation by senior Lockheed Martin personnel to the Defense Acquisition Performance Assessment Panel shows that in today's environment, a typical 40-year career span encompasses far fewer military aircraft programs than in generations past. In the 1950s, the aerospace engineering industry would have exposed a new engineer to 84 different aircraft over a 40 year period. An aerospace engineer starting out in the year 2000 might see only a handful during a 40 year career [6].

TABLE 1
Failure Types

	Exposure To Loss	**Opportunity To Learn**
Optimal	Low	High
Epic	High	Low

In a similar trend, the number of aerospace companies available to choose from has decreased significantly. For example, in 1970, when the A-10 was being developed, "twelve companies were selected to receive the RFP… six companies responded with proposals." [5]. Today, the Air Force is basically limited to three aircraft companies: Lockheed-Martin, Boeing and Northrop. When it comes to building manned jet fighters, these companies are more likely to cooperate than compete, which is a mixed blessing at best.

This decrease in the number of aircraft being developed (and in the number of aircraft developers) has a negative effect on the industry's overall experience level. Unlike the military aerospace community in the 1950's and 1960's, today's aircraft engineers, program managers and other project leaders have far fewer opportunities to gain experience and perspective, to experiment and learn, or even to talk with other people whose experiences in manned aircraft design diverge from their own. This situation is not limited to military aircraft, as many defense related fields are tending towards producing small numbers of big projects rather than large numbers of smaller projects. A similar case could easily be made for C4ISR systems, tanks, artillery or naval ships. This reduced exposure to different projects provides limited opportunities to learn, which has a negative impact on the community's overall performance.

However, the situation with Unmanned Aerial systems (UASs), which perform a significant percentage of airborne ISR missions, is far different. In fact, the UAS development environment appears to be the inverse of the manned aircraft situation. The result is a wide range of opportunities for experimentation and learning among developers, while simultaneously providing innovative, world-class capabilities for the US military across a broad spectrum of operations.

Thus, the concern about the decrease in new manned aircraft development is partly allayed by the large number of unmanned aircraft currently being developed. This pattern bodes poorly for innovation and growth in manned aircraft, but indicates the future of UAS's is likely to be marked with further breakthrough capabilities and discoveries.

There are several reasons UAS developers are able to provide such an impressive variety of platforms. The most obvious is the immature state of UAS applications – it is still a relatively new field, with many niches unfilled and opportunities unmet. Unlike the more mature manned fighter or bomber communities (for example), we are still learning what UAS's can do and have not yet fully established their operational boundaries, or even the optimal doctrines to guide their use.

Even a cursory survey of UAS's currently in operation reveals a wide variety of body shapes, sizes, missions, payloads, endurances and capabilities. This diversity is made possible because the underlying technology is relatively low-cost, simple and can be rapidly developed and deployed by small teams. This also fosters competition among developers, which has the potential to convey significant benefits for the DoD in terms of cost, availability and performance.

IV. IMPLEMENTING FIST

There are a number of things project leaders can do to implement the FIST approach, beginning with recognizing that FIST is one unified idea, not four separate ideas. The four components of FIST are connected in many ways, but most prominently by a common thread of simplicity. Accordingly, project leaders seeking to express the FIST value set should emphasize simplicity.

Second, project leaders need to genuinely understand the values and avoid settling for ineffective, superficial imitations. It is not adequate to establish the mere appearance of speed instead of actually delivering operational capabilities on a short timeline. FIST also requires a certain tolerance for failure, and project leaders should expect to pursue multiple iterations. Finally, the FIST value set is most effective if the values are shared by all the stakeholders, so project leaders are advised to explicitly discuss the concept with their team, their customers and their suppliers.

For C4ISR system development projects, FIST advocates decomposing the domain into requirement sets that can be iteratively addressed in FIST-sized efforts, then fielding capabilities with integration points baked in. This ensures timely delivery of modular components, avoiding the dual dangers of technologies that are obsolete and / or stovepiped. The operational environment for C4ISR systems is increasingly multi-service and multi-national, making simplicity and integrability more important than ever.

A. FIST Guidelines

There are many ways to express the FIST values, depending on factors such as operational context, type of technology, and customer. In some cases, industry has created very robust methodologies based on the FIST values, as with the Agile software development approach. The Defense Department has similarly produced a variety of rapid development processes, including the Joint Urgent Operational Need, the Warfighter Rapid Acquisition Process and the Air Force Research Laboratory's Innovate Solutions To Urgent Needs process. These existing processes and methodologies should be leveraged as much as possible when developing new C4ISR systems.

Table 2 provides a short collection of actionable guidelines for project leaders seeking to use FIST.

TABLE 2
FIST GUIDELINES

1.	Discuss values with team, stakeholders, customers and suppliers
2.	Minimize team size, maximize team talent.
3.	Use schedules and budgets to constrain the design.
4.	Insist on simplicity in organizations, processes and technologies.
5.	Incentivize and reward under-runs.
6.	Write requirements that are achievable within short time horizons.
7.	Produce designs that only include mature technologies.
8.	Documents and meetings must be short. Have as many as necessary, as few as possible.
9.	Delivering useful capabilities is the only measure of success.

B. FIST Heuristics

Program management and systems engineering are primarily social disciplines, not hard sciences. They involve subjective questions of morale, opinion, values and other intangibles, and their secrets are generally not revealed through statistical analysis, however complex. Attempts to treat system development methodologies as optimizable mathematical operations are inappropriate and tend to produce unhelpful solutions or irrelevant insights, based on an illusion of rationality.

To borrow a phrase from Nicholas Taleb, the mathematical approach treats program management like "a second-rate engineering problem for those who want to pretend that they are in the physics department." [9]. The truth is, project leaders do not deal solely with questions of physics and mathematics. Accordingly, the project leader's tool kit should include general guidelines and principles, and not solely rely on formal rules and equations.

Heuristics, therefore, are popular tools for disciplines like systems engineering, systems architecture and program management, where practitioners confront complex problems and unique situations. Table 3 provides a collection of FIST-based heuristics. These rules of thumb are intended to illuminate facets of the systems engineering art, to guide system developers, to encourage what Schön calls "reflective practice," [8] and to offer memorable summaries of reliable principles.

It must be noted that heuristics are abstractions of experience, not mathematical certainties. They are general principles, not hard-and-fast rules. They should be used as vectors, not boundaries, and treated as reliable recommendations, not commands. Like zen *koans*, they are for pondering, for thoughtful consideration and experimentation. Bear in mind the heuristics listed below are "not proven by the stories, but are designed to be general truths that are illustrated by the stories." [7].

The objective of this list is to expand what Schön calls the practitioner's "*repertoire* of examples, images, understandings and actions." [8, *emphasis in original*]. Under Schon's methodology, practitioners use their repertoire to understand unfamiliar problems in terms of familiar ones, and unsolved

Table 3
FIST Heuristics

1.	A project leader's influence is inversely proportional to the project's budget and schedule.
2.	Creative constraints foster creativity. Adding time and/or money generally does not improve outcomes.
3.	Fixed funding and floating requirements are better than fixed requirements and floating funding.
4.	Complexity is a cost.
5.	Complexity reduces reliability.
6.	Simplicity scales. Complexity doesn't.
7.	An optimal failure costs a little and teaches a lot. When FIST projects fail, they fail optimally.
8.	Iteration drives learning, discovery and efficiency. FIST is iterative.
9.	Talent trumps process.
10.	Teamwork trumps paperwork.
11.	Leadership trumps management.
12.	Trust trumps oversight.

problems in terms of solved ones. Expanding the repertoire puts more tools at the practitioner's disposal, enabling the community to understand and solve problems which were previously unfamiliar or unsolved.

For example, heuristic #6 states "Simplicity scales. Complexity doesn't." This might be a useful guide for project leaders when faced with a complex operational environment and a wide range of required capabilities. In such a situation, operators and engineers alike might be tempted to respond to the complexity with a complex solution. This approach would ultimately limit the system's availability and utility, because of the cost, delay and effort involved with scaling up (see previous comments about the state explosion problem). In what may initially seem a counterintuitive move, simplifying the system actually facilitates a larger-scale application.

V. CONCLUSION

Every decision project leaders make, from organizational structure to architecture design, is a reflection of their values. These values have a significant impact on the project's objectives and measures of merit, and on the system's operational effectiveness once it is fielded.

When measures of merit are defined in terms of how quickly and inexpensively the system can be delivered, how simply it can be operated and maintained, and how small its logistical and physical footprint is, the project leaders are expressing the FIST value set.

Project leaders who accept the FIST values tend to establish streamlined organizations which rapidly deliver simple, inexpensive and effective systems, using straightforward processes and procedures. On the other hand, project leaders who value complexity establish byzantine enterprises and build complicated systems. Those who think a large budget and a long schedule are signs of a healthy project end up spending a lot of time and money.

The FIST value set is not the only constructive set of values for C4ISR system development projects, nor does it guarantee a successful outcome. However, the FIST values convey four significant benefits to C4ISR systems: stability, accountability, learning and optimized failures. FIST is therefore recommended as a constructive set of values for system development projects. The guidelines and heuristics in this paper provide specific recommendations for understanding and applying this approach.

NOTE: Portions of this paper are excerpted from a thesis entitled "The Effect of Values on System Development Program Outcomes," number AFIT/GSE/ENV/09-M08, completed at the AF Institute of Technology, Mar 2009.

I. REFERENCES

[1] Air Force Studies Board, *Pre-Milestone A and Early-Phase Systems Engineering: A Retrospective Review and Benefits for Future Air Force Acquisition*. Washington D.C.: The National Academies Press, 2008.

[2] Christensen, D., Searle, D., Vickery, C. "The impact of the Packard Commission's recommendations on reducing cost overruns." *Acquisition Review Quarterly*: 251 – 262 (Summer 1999)

[3] Clark, Asa A. *The Defense Reform Debate: Issues and Analysis*. Baltimore: The Johns Hopkins University Press, 1984.

[4] Fitsimmons, J. and van Tol, J. "Revolutions in military affairs." *Joint Forces Quarterly*, 4:24-31 (Spring 1994).

[5] Jacques, D. & Strouble, D. "A-10 Thunderbolt II case study." Air Force Center for Systems Engineering, 2008.

[6] Kubasik, C. and Heath, R., *Lockheed Martin Perspective on Improving the Defense Acquisition System: Before the Defense Acquisition Performance Assessment Panel*, Aug 17, 2005.

[7] Laufer, A. *Project Management Success Stories*. New York: Wiley, 2000.

[8] Schön, D. *The Reflective Practitioner*. Boston: Basic Books, 1984.

[9] Taleb, N., *The Black Swan*. New York: Random House, 2007.

[10] Ward, Daniel. *The Simplicity Cycle*. Rogue Press, 2005.