SYM-AM-16-027

PROCEEDINGS
OF THE
THIRTEENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM

WEDNESDAY SESSIONS
VOLUME I

Acquisition Cycle Time: Defining the Problem

David Tate, Institute for Defense Analyses

Published April 30, 2016

Prepared for the Naval Postgraduate School, Monterey, CA 93943.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL

The research presented in this report was supported by the Acquisition Research Program of the Graduate School of Business & Public Policy at the Naval Postgraduate School.

To request defense acquisition research, to become a research sponsor, or to print additional copies of reports, please contact any of the staff listed on the Acquisition Research Program website (www.acquisitionresearch.net).

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL

Panel 4. Strengthening Schedule Estimates in MDAPs

Wednesday, May 4, 2016	
11:15 a.m. – 12:45 p.m.	**Chair: Rear Admiral Thomas J. Kearney, USN,** Director, Acquisition, Commonality, & Expeditionary Warfare, Naval Sea Systems Command ***Acquisition Cycle Time: Defining the Problem*** David Tate, Institute for Defense Analyses ***Schedule Analytics*** Jennifer Manring, Principal Economics and Business Analyst, The MITRE Corp. Thomas Fugate, Principal Acquisition and Program Management Analyst, The MITRE Corp. ***Toward Realistic Acquisition Schedule Estimates*** Raymond Franck, Professor Emeritus, U.S. Air Force Academy Gregory Hildebrandt, PhD Bernard Udis, Professor Emeritus of Economics, University of Colorado at Boulder

Acquisition Cycle Time: Defining the Problem

David M. Tate—joined the research staff of the Institute for Defense Analyses' Cost Analysis and Research Division in 2000. Since then, he has worked on a wide variety of resource analysis and quantitative modeling projects related to national security. These include an independent cost estimate of Future Combat Systems development costs, investigation of apparent inequities in Veterans' Disability Benefit adjudications, and modeling and optimization of resource-constrained acquisition portfolios. Dr. Tate holds bachelor's degrees in philosophy and mathematical sciences from Johns Hopkins University, and MS and PhD degrees in operations research from Cornell University. [dtate@ida.org]

Abstract

Acquisition cycle time—roughly speaking, the development lead time to field a working system once we have identified a need for a new materiel solution—is a hot topic in defense circles. This work takes a data-driven look at historical and current acquisition program cycle times. We show that the trend toward longer cycle times over the past few decades is restricted to a handful of high-profile programs, which has profound implications for effective policy response. We also show evidence that cycle times are driven by system complexity, and that schedule slip is associated primarily with overly optimistic schedule estimates. We conclude with a discussion of the increasing importance of software for development lead times, and the general problem of trading capability for timeliness in acquisition programs.

Introduction

The Cycle Time Problem—Perception

Cycle times (or development lead times) for defense weapon systems are a hot topic in the defense world. In September 2014, the Under Secretary for Acquisition, Technology, and Logistics, the Honorable Frank Kendall, issued Version 3 of the Better Buying Power (BBP) initiative. One focus area of BBP 3.0 is to "reduce cycle time while ensuring sound investments" (Kendall, 2014). The lead article (Schultz, 2014) in the November–December 2014 issue of *AT&L Magazine* was entitled "Please Reduce Cycle Time." RAND produced a 2014 report entitled Prolon*ged Cycle Times and Schedule Growth in Defense Acquisition: A Literature Review* (Riposo, McKernan, & Kaihoi, 2014). The House Armed Services Committee held hearings in October 2015 on the topic "Shortening the Defense Acquisition Cycle."

Many commenters on defense acquisition have asserted that it now takes too long to develop and field new systems, and that we should do something about it. They say that the pace of technological advancement, especially in electronics and information technology, is now so fast that our "advanced" military systems are nearly obsolete by the time they are fielded. Even seemingly mundane systems, such as troop transport vehicles and cargo aircraft, seem to take forever to field.

The various stakeholders in the defense community have put forward numerous theories for what is causing the problem, many of which place the blame squarely on someone else. Worse yet, the appropriate policy response to the cycle time problem depends sharply on which theory one subscribes to. Some observers diagnose excessive oversight and prescribe a more laissez-faire approach to acquisition. Others diagnose unaffordable ambitions and unnecessarily demanding requirements, and prescribe appetite suppression and fiscal discipline. Still others diagnose inept management and excessive bureaucracy, and prescribe streamlined processes and organizations.

The Cycle Time Problem—Reality

In this paper, we will address several important questions related to the cycle time problem. First, we will consider empirically whether there is a general cycle time problem across all programs, or a specific cycle time problem within a few programs. Second, we will look at some candidate reasons why development takes as long as it does. Finally, we will consider the overarching question of how to decide what to try to buy, and how to try to buy it, given an understanding of how long it is likely to take.

Cycle Times for Typical Programs Are Not Increasing

Figure 1 shows a scatter plot of cycle times[1] for Major Defense Acquisition Program (MDAP) subprograms (including initial development) as a function of when each system reached its Initial Operational Capability (IOC), or equivalent. Going back to the late 1980s, there is no apparent upward trend. Statistical analysis confirms that the trend is indistinguishable from zero, and that the median cycle time has been roughly eight years over that entire span. This absence of trend in the median holds for all commodity types—aircraft, ground systems, space systems, ships, etc.

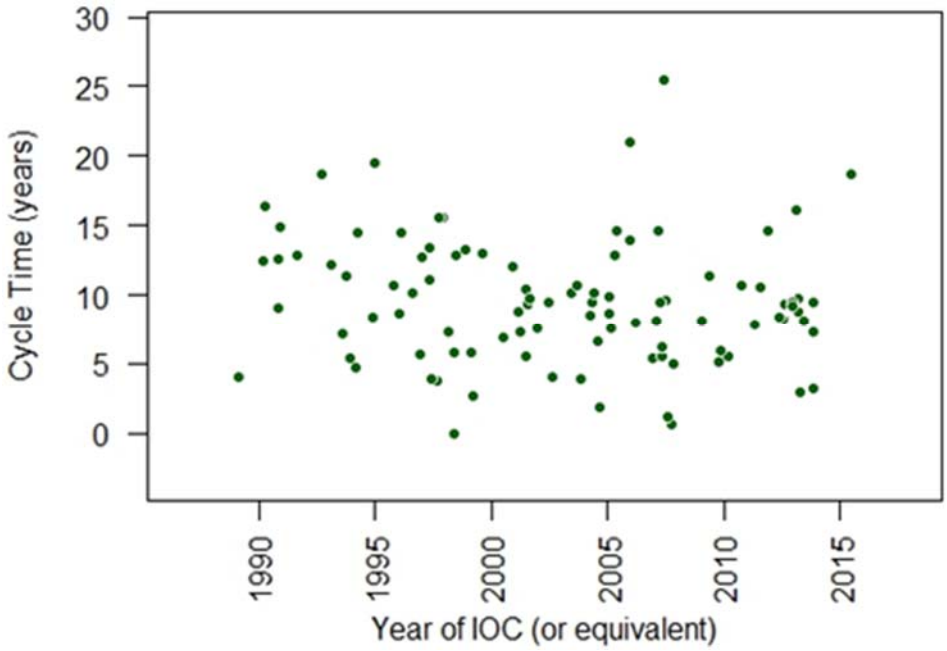

Figure 1. Program/Subprogram Cycle Time by IOC Year

Why, then, is there a perception that cycle times have been getting worse and worse? Figure 2 shows the same data, but with each point now proportional to the final (or

[1] *Cycle time* is defined here as the time in years from program initiation to IOC or equivalent. For most programs, initiation is at or just before what is now called Milestone B. For some programs, initiation is at Milestone (MS) A, or even earlier if the program began as a Technology Demonstration program. For programs with no formal IOC, the equivalent milestone might be OPEVAL, First Unit Equipped, etc. The plot shows cycle times for ~100 subprograms for which both cost and schedule data were available.

most recent estimated) total procurement cost of the system being developed, in inflation-adjusted dollars.

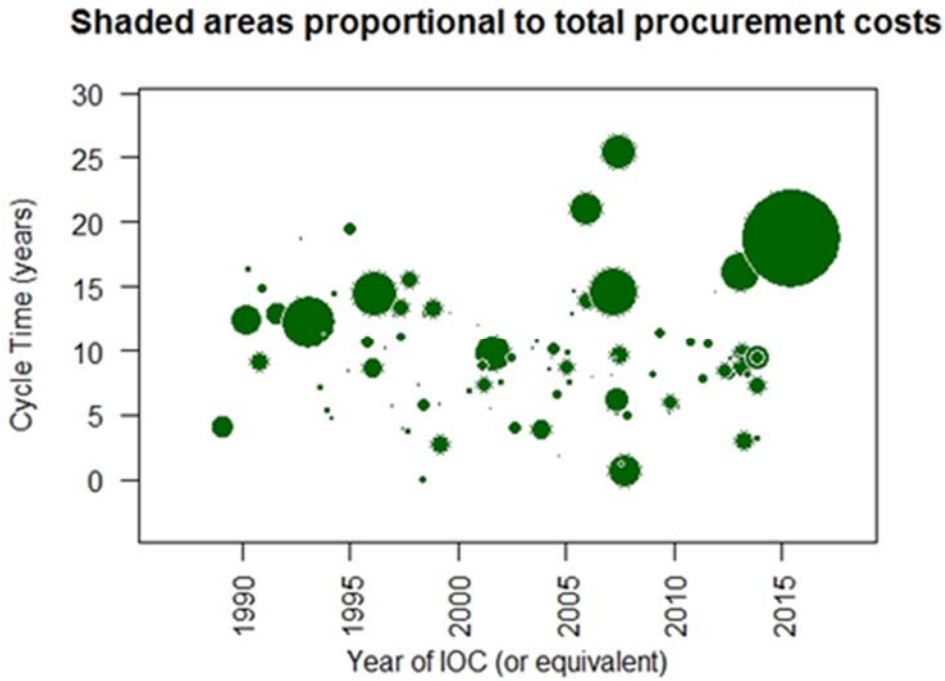

Figure 2. Cycle Time by IOC Year Showing Relative Program Size

Here, there is a noticeable upward trend for the programs that are spending the most money on procurement. The cycle times for these highly visible programs may be driving the perception that things are taking longer.

Cycle Time Growth Is Getting Worse for Some Commodity Types

The sense that things are taking longer is driven not just by actual cycle times, but also by cycle time growth, or "schedule slip." Even where programs are not taking any longer than they previously did, the Office of the Secretary of Defense (OSD) and Congress notice when programs take longer than promised.

Even here, there is no significant overall statistical trend in program schedule growth over the past 25+ years. Figure 3 shows the relative schedule growth of each of the subprograms portrayed in Figure 1 and Figure 2, broken out by commodity type. Unlike the overall cycle time story, here there are significant differences among commodity types. Overall schedule growth trends are flat or downward (with occasional outliers) among ground systems, aircraft, missiles, and ships. The trend is upward for space systems and Command, Control, Communications, & Intelligence (C3I) systems. In addition, instances of individual programs with schedule growth well above the overall trend for their commodity type are becoming more common.

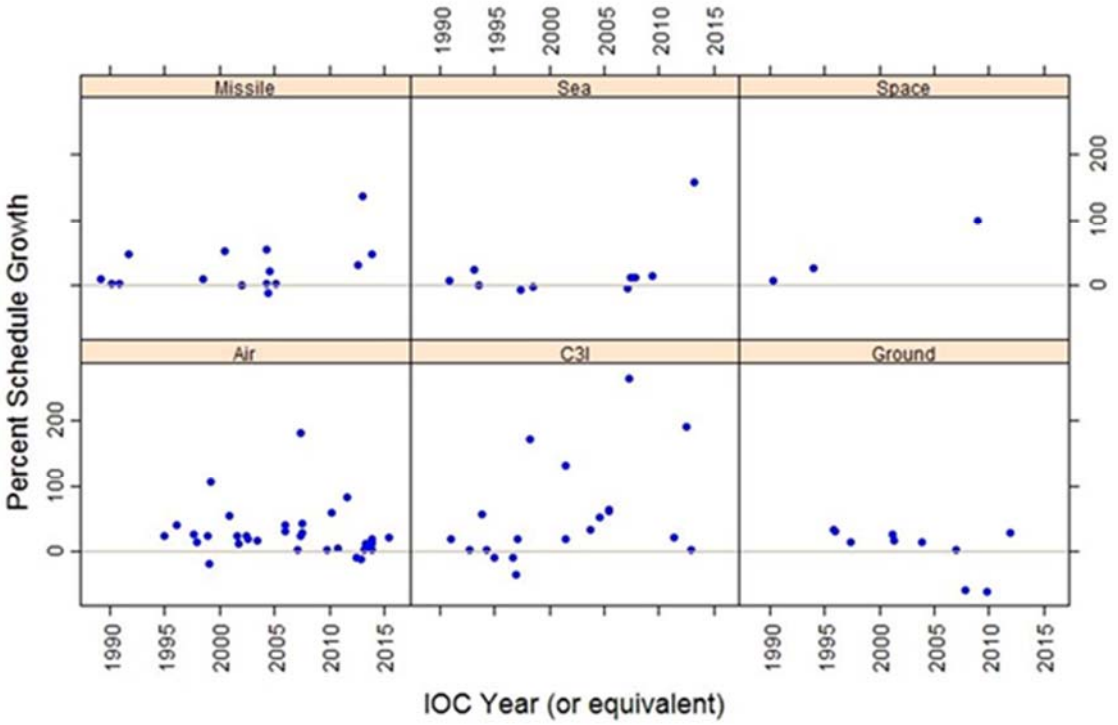

Figure 3. Percent Schedule Growth by IOC Year for Various Commodity Types

Of course, schedule growth does not necessarily indicate poor program execution. It may be that the program's funding was cut. It may be that requirements were changed. Or, it may be that the original schedule estimate was unreasonable.

We computed a measure of *relative schedule optimism* for MDAPs, defined as the difference between the average cycle time for programs of the same commodity type and the given program's estimated cycle time at program initiation, divided by the commodity average. Thus, for example, a program forecast to take six years when the average for its commodity type is eight years would have a relative schedule optimism of (8–6)/8 = 25%. Larger numbers indicate more optimism; negative numbers indicate pessimism relative to the average. Figure 4 shows a graph of cycle time growth versus this metric. We see that a clear relationship exists between schedule optimism and schedule growth for both new start programs and modifications of existing systems. Interestingly, the average percent schedule growth for a given level of optimism is greater than the amount of optimism. This suggests either that excessive optimism is a symptom of a deeper problem, or that there are cascading effects from being too optimistic. We will return to that question when we discuss software development in the section titled The Special Case of Software.

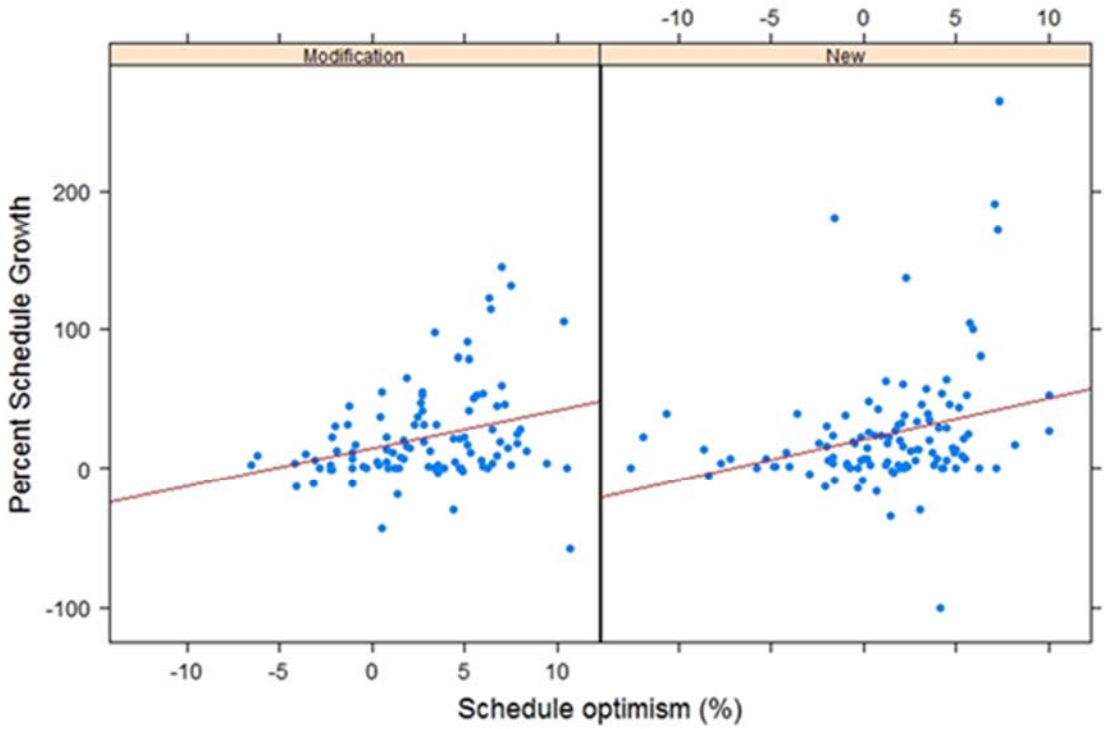

Figure 4. Relationship Between Schedule Optimism and Schedule Growth

Do Cancelled Programs Make the Picture Look Better or Worse?

A confounding factor in any study of MDAP cycle times has been the high rate of program cancellations over the past 15+ years. As famously reported in the Decker-Wagner report (2011),[2] between 1995 and 2009, the Army spent roughly one quarter of its Development, Test, and Evaluation (DT&E) funding on programs that delivered fielded capability. The other Services were by no means immune to this problem.

Cancelled programs bias the statistics on cycle time by censoring data from programs that would tend to take longer than average if carried to completion. The effect of individual cancellations on average or median cycle time depends on why the program was cancelled. Programs that were cancelled because they were going to be obsolete by the time they were fielded[3] should have their expected durations included in the distribution of cycle time outcomes if our goal is to understand the extent to which programs take too long to be relevant or timely. Programs that were cancelled for other reasons—for example, technical infeasibility or unaffordability—do not generally tell us anything about achievable cycle times for executable programs.

---

[2] The actual title of this report is *Army Strong: Equipped, Trained and Ready: Final Report of the 2010 Army Acquisition Review*. It is often referred to as "the Decker-Wagner report," after its two co-chairmen, Gilbert F. Decker and Louis C. Wagner.

[3] Obsolescence can be due to changes in the threat environment, geopolitical events, new technologies, etc.

However, cancelled programs are certainly relevant to the larger question of how long it takes to mitigate a capability gap, once it has been identified. Of the major programs cancelled since 2000, few if any were cancelled solely due to premature obsolescence. In most cases where obsolescence was cited by OSD or the Service as a causal factor, the program was also technically infeasible or unaffordable. It is not unreasonable to think of the current Amphibious Combat Vehicle (ACV) program as the direct continuation of the earlier Expeditionary Fighting Vehicle (EFV) and Advanced Amphibious Assault Vehicle (AAAV) programs. By that measure, the Marine Corps has been attempting to acquire a long-range high-speed amphibious troop carrier since 1995, and does not expect to be able to field one until 2025 at the earliest. In the same way, the Army's ongoing attempts to replace the OH-58 Kiowa scout helicopter began in 1985 with the Comanche program. Each Service has outstanding modernization requirements left unfilled by failed programs; not all of those requirements have any current MDAP attempting to meet them. The delays in fielding these capabilities may have operational impacts, but they do not seem to result from inefficient "acquisition" in the usual sense.

Execution or Expectations?

The historical cycle time data raise some important questions. First, do long cycle times reflect an acquisition process problem (to be addressed by changes to the acquisition system) or an outlier problem (to be addressed by root cause analysis and improved oversight)? It would probably be neither efficient nor effective to overhaul the entire acquisition system if most programs are executing reasonably under the current system.

Second, is this a problem of execution or of expectation? How long *should* it take to develop a fifth-generation fighter aircraft, or a first-ever tilt-rotor transport aircraft, or a high-speed amphibious assault vehicle? Were the original schedule estimates implausible? Where do the development schedules found in Acquisition Program Baselines come from, anyway?

Finally, to what extent are our acquisition processes the pacing factor in MDAP development? Are there unnecessary regulations that slow down development without adding value? Are there unnecessary administrative processes (either within OSD or within the Services) that delay development? Is testing a cause of delay, or merely the bearer of bad news? Are there technical reasons why we should not expect to be able to go much faster than we currently do?

Which Came First—The Program or the Schedule?

When deciding which new programs to start and what kind of system they should aim to produce, decision-makers are informed by cost and schedule estimates. The National Air and Space Administration (NASA) goes so far as to treat cost and schedule jointly, recognizing both that they are highly correlated and that changes driven by one will affect the other. No one is surprised when the cost estimates are based on the content of the program—the capabilities the system is supposed to provide, the materials it will use, the maturity (or immaturity) of the technologies to be employed, etc. OSD performs an independent cost estimate for all major acquisitions, which is taken seriously during milestone deliberations and tends to offset some of the more optimistic tendencies of the sponsoring Services.

There is no corresponding attention paid to OSD concerns about schedules, however. Not infrequently, the initial schedule *estimate* for an MDAP is not an estimate at all, but a constraint set externally with little regard to program content or historical precedent. Sometimes this is driven by anticipated external demands for a system that is to

be used on multiple platforms, as was the case for several of the Joint Tactical Radio System (JTRS) subprograms. Sometimes it is driven by a planned retirement agenda for existing systems, such as the plan for the Global Hawk Block 30 aircraft to replace the U-2. Sometimes it seems to be driven by impatience; the Army's never-quite-started Ground Combat Vehicle program was told the delivery date of the first production vehicle in its Initial Capabilities Document before even a design concept had been identified.

How Long Should It Take?

No one would argue with the claim that a cost estimate should be based on the content of the program, or that it should be informed by the history of past efforts to develop similar things. Surprisingly, the analogous argument for schedules gets much less traction. The link between program content and development cycle time has not been as clearly established among decision-makers. There is a lingering suspicion that better management, less red tape, or less oversight would allow successful completion of development projects much more quickly than they have been done in the past.

There have been some substantive studies that looked at this question in specific domains. Among the most comprehensive is Gene Bearden's work at the Aerospace Corporation on the cost and schedule drivers of space probes. Dr. Bearden developed a sophisticated complexity metric for space probe projects, based on the technical details of the operational domain (earth orbit or planetary), the propulsion technology, the required data links, the payload instruments, etc. He then showed that there is a powerful and consistent relationship between project complexity and development cycle time. More importantly, nearly all partial or complete mission failures occurred when NASA attempted to develop and launch probes more quickly than the estimated required development time. Figure 5 shows this relationship.

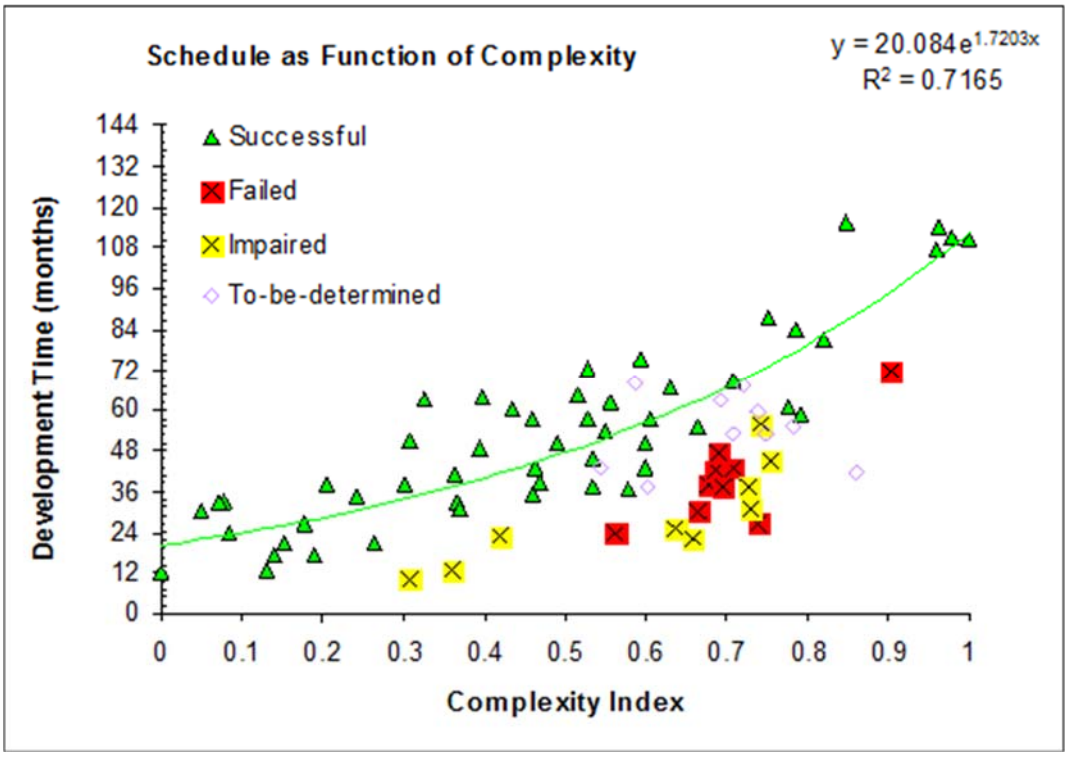

Figure 5. Relationship Between Space Probe Complexity, Development Time, and Mission Success
(Courtesy of and reprinted by permission of the Aerospace Corporation)

We also investigated whether MDAP cycle times are related to either the number of Critical Technology Elements (CTEs) identified in the program's Technology Readiness Assessment (TRA), or the number of distinct program requirement records in the "Performance" fields of the Defense Acquisition Management Information Retrieval (DAMIR) system. Both are positively correlated with cycle time, although we did not have CTE data for enough programs to establish statistical significance. For requirement counts, there is a strongly significant relationship with cycle time during times of growing defense budgets, and no relationship at all in times of decreasing budgets.[4] These results taken together suggest that schedule is driven by program content, but that programs perhaps shed requirements in times of tight budgets.

The Special Case of Software

The connection between complexity and cycle time is also well understood in the software industry. In his pioneering book *The Mythical Man-Month,* Fred Brooks (1975) presented a few key facts about software development projects:

[4] Given that the raw number of requirement records in DAMIR is only weakly correlated with actual program complexity, the existence of a strongly significant (p = 0.0002) relationship between requirements records and cycle time is somewhat surprising. The fact that this relationship is only apparent during times of growing budget is even more surprising. It is possible that this metric is measuring bureaucratic complexity more than technical complexity.

- Adding staff to a software project that is behind schedule makes it take longer.
- There is a lower bound on the number of defects in a software system. The more complex the system, the higher this bound.
- Software development consists of completing known work and discovering new work. There are fundamental limits on the efficiency of both aspects.
- The duration of a software development project depends strongly on the degree of coordination required among the various software modules.

Lawrence Putnam (1978) used analogous reasoning about completion and discovery to derive a quantitative model of project duration. He found that total development cost is a function of schedule—but not in the expected sense that a longer project costs more. Rather, there is a natural "most efficient" schedule for a given set of requirements, and any attempt to accelerate the development to finish more quickly than that natural duration results in increased cost. Worse still, complex projects are only slightly compressible; there is a sharp asymptotic limit to how fast you can try to complete the project without breaking it. Figure 6 shows this relationship schematically. For schedules longer than the natural schedule, cost increases roughly linearly with duration due to low staff utilization, as shown by the dashed line. For schedules shorter than the natural schedule, staff utilization is high, but completion outpaces discovery, leading to inefficient rework and low quality. Schedules significantly shorter than the natural schedule are simply not possible, and development efforts attempting to go faster than that generally fail.

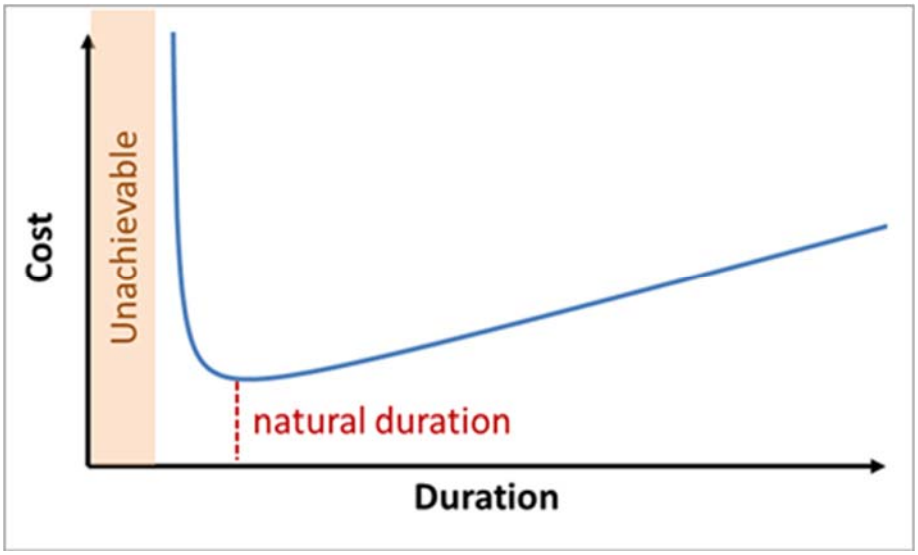

Figure 6. Cost as a Function of Duration for a Given Set of Software Requirements

Are Weapons Systems Like Software Systems?

We have shown that feasible schedules for space systems and for software seem to be constrained by program complexity. Is this true for all weapon systems? If not, which systems might it be true for?

Clearly, complexity is not a factor when buying truly commercial items. By extension, complexity should not be a factor when buying systems that are minor modifications of commercial items, or that use only established commercial technologies. This is at the heart of the Government Accountability Office's (GAO's) recurring admonition that acquisition best

practice requires that all critical technologies for a program be mature[5] before MS B. It is also central to the Decker-Wagner report's taxonomy of acquisition risk categories, with corresponding development timelines. In that report, a low-risk acquisition is defined to be one that purchases an existing commercial item or modifies an existing system. A moderate-risk acquisition is defined to be acquisition of a system that uses only mature, proven technologies. Even then, the report cautions that you should expect 6 to 11 years from Materiel Development Decision (MDD) to MS C if you are developing a new system that does not use exclusively pre-existing components (p. 99).

However, maintaining our technological military advantage cannot be accomplished by only buying things that already exist. The Decker-Wagner report advises the Army to manage risk in its acquisition portfolio by limiting the proportion of higher-risk programs to "only those [systems] that are truly urgently needed because they represent 'game-changing,' revolutionary military capability, e.g., atomic bomb, night vision, fire-and-forget missiles, and stealth" (p.106). It also cautions that you should expect an 8–14-year development cycle (MDD to MS C) for such systems, even if you do everything right.

Are Weapon Systems Actually Software Systems?

There is another, more compelling reason to think that weapon system acquisition programs might behave like software development programs—namely, that they are software development programs. Nearly every MDAP today involves more software than even the most software-intensive programs of 20 years ago. The F-35 aircraft system, in the culmination of a trend begun more than 50 years ago, has almost no functions that are not implemented, mediated, or controlled in software.

We have noted that there is a natural minimum duration for a software development program. Even if it were true that hardware development and integration are no harder today than they have been in the past, we would expect there to be a size of software effort at which the software development portions of the program begin to dominate the schedule. Historically, the critical path for system development has run mostly through the hardware side of the project. Software contributed vital capabilities, but only a small portion of the program cost or duration. As we design systems that perform more and more of their functions using software, that might no longer be true.

In fact, there is some evidence that we may already be reaching the turnover point where software development drives schedule duration. To test the plausibility of this idea, I used the COCOMO-II software effort estimation tool to estimate the duration of large, difficult (but not too difficult) software development projects, as a function of Source Lines of Code (SLOC). I then collected data on the cycle times of recent software-intensive MDAPs, along with their SLOC at IOC. The results are shown in Figure 7.

[5] The GAO defines maturity, for this purpose, as a Technology Readiness Level (TRL) of 6 or higher. TRL 6 is defined as successful demonstration of a representative prototype operating in an environment that is operationally relevant, given the system requirements.

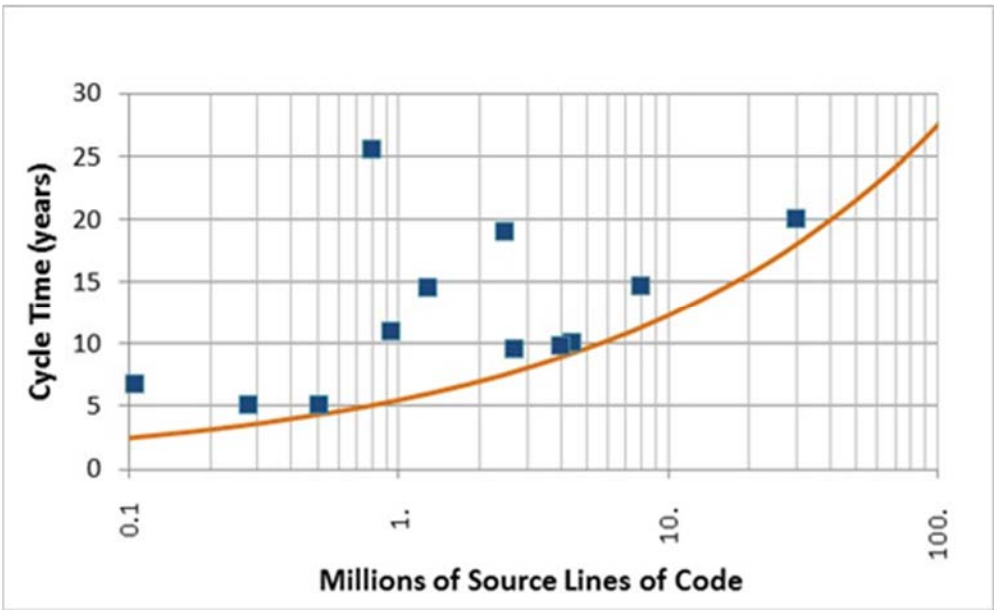

Figure 7. Development Times of Recent MDAPs vs. Software Size at IOC

The COCOMO-II prediction curve shown here is notional. COCOMO-II takes as input up to 24 separate "effort multiplier" parameters and five "scale factors" related to economies (or diseconomies) of scale for software projects. The curve shown here corresponds to a software project that is modestly above average in all dimensions.[6] It is intended to provide an empirically based estimate of the proper scaling of duration as a function of size for software similar to the kind found in Department of Defense (DoD) weapon systems. Astonishingly, without any adjustment, these initial parameter choices produce a curve that seems to behave like a tight lower bound on cycle time for the systems in question. This suggests that even if software is not already defining the lower bound for MDAP cycle times, it soon will.

Again, the space systems community may be slightly ahead of the DoD in reaching this conclusion. In a 2009 presentation, Dr. Steve Jolly (2009b) of Lockheed Martin Space Systems concluded,

> Software/firmware can no longer be treated as a subsystem. Systems engineering organizations need to engineer the software/avionics system—a change in leadership technical background. … The game has changed in developing space systems. Software and avionics have become the system [emphasis in original]. Structures, mechanisms, propulsion, etc. are all supporting this new system.

[6] For those familiar with COCOMO-II modeling, the curve shown in Figure 7 corresponds to an Effort Multiplier (EM) of 1.5 and an exponent (E) of 1.1. The EM reflects a development only slightly higher than nominal in difficulty or complexity in all dimensions. The exponent corresponds to a total Scale Factor (SF) of 18, which is very close to the SF value for a "nominal" development.

In a related article for NASA's *ASK Magazine*, Jolly (2009a) speculates that this is because software now touches every part of the system. It is no longer an isolatable subsystem or module that can be designed in parallel with the system; it is the common element that ties together all parts of the system. The software is the integrating element, and as such must be the central feature of the design.

Not all weapon systems are as software-intensive as a NASA space probe—but many of them are. In particular, when the role of software in the program is no longer such that the software can be treated as a separable module, but rather as an integrative framework, it will be necessary to manage the program as a software development project with associated hardware and cyber/physical integration, rather than as a hardware development project with associated software.

Acquisition Implications

What Can Be Had Quickly?

Putting together the data regarding system complexity, software content, and technology maturity, we can see that acquisition cycle times are bounded below by the maximum of several possible limiting factors. If the system is technically immature, we will not be able to field it very quickly. If the system involves very large amounts of new software, or highly integrated software and hardware, we will not be able to field it very quickly. This would be true even with ideal program management, ample and stable funding, and Acquisition Reform that eliminates all red tape and oversight.

So what can we get quickly?

1. Truly non-developmental items—commercial products and systems that already exist. (Think MRAP.[7])
2. Upgrades of existing systems that insert already-mature technology and do not overstrain the size, weight, power, and cooling capacity of the current platform. (Think UH-60M or M1A2 Abrams upgrades.)
3. Integration of existing mature systems into new capabilities. (Think HIMARS.[8])
4. New systems developed using agile methods, in which users (user representatives) work interactively with developers to identify and evolve a set of capabilities that are useful enough to be worth fielding, rather than working toward pre-set Threshold and Objective requirements. (Think CREW.[9])
5. New systems with extremely limited requirements, where we are willing to live with capabilities at or below current systems in most dimensions, in order to get enhanced capabilities in one or two urgently needed dimensions. (Think F-117A.)

---

[7] Mine-Resistant Ambush-Protected family of vehicles.
[8] High-Mobility Artillery Rocket System, derived by mounting a Multiple Launch Rocket System (MLRS) launcher on a 5-ton Family of Medium Tactical Vehicles (FMTV) truck.
[9] Counter-radio-controlled Explosive Device program; a cumulative series of technology deployments including (for example) the Thor III man-portable jammer.

6. New systems whose critical technologies and basic operational characteristics have already been demonstrated at TRL 5 or higher in *push* research and development or demonstration programs. (Think Predator.)
7. New modular subsystems that can be used to replace older, less-capable subsystems on platforms that have both modular architectures and sufficient design margin (e.g., size, weight, power, cooling, etc.) to be able to accommodate and integrate the new technology. (Think DDG-51 Modernization.)

Of these approaches, only the final three or four have the potential to sometimes produce leap-ahead technology advantage. Each of those, in turn, has its limitations and caveats.

In the case of "limited requirements" developments, operational effectiveness is liable to be short-lived, requiring more deliberate follow-on programs that incorporate the leap-ahead technology in a more well-rounded package. In the case of R&D *push* from technology demonstration programs, someone has to have had the foresight to fund many research and development programs (not associated with major systems procurement) in the relevant technology areas over the preceding decade, so that there are mature technologies on the shelf to choose among. Even then, the initial MDAP incorporating those technologies is liable to produce a partially successful and fragile initial solution that will need to be followed up with more robust designs, as was the case with Predator.

Rapid incorporation of new modular subsystems on an existing platform depends on the existence of a robust, overdesigned, modular-architecture platform that can host the upgrades. Initial development and deployment of those host platforms will typically not be especially rapid, and their initial capabilities may not be significantly better than legacy platforms. Their value is in their ability to enable future upgrades (see, for example, Patel & Fischerkeller, 2013).

Finally, in the case of agile development, you would have to get very lucky to produce a leap-ahead capability. Most iterative agile developments will produce useful and timely (but not revolutionary) capabilities.

When Should an Analysis of Alternatives Consider Cycle Time?

The most important decision in any acquisition is the choice of what to buy. The purpose of the Analysis of Alternatives (AoA) is to provide decision-makers with all of the relevant information regarding the cost, schedule, and effectiveness of each of the available alternatives—including a characterization of the risks in each of those dimensions. The worst acquisition failures are the result of choosing an alternative that is not physically possible, or is unaffordable, or has very little chance of being delivered in time to be useful.

Historically, schedule has not generally been considered as either a consequence of the choice of alternative, or as a Key Performance Parameter for the program. In some cases, this makes perfect sense—most any alternative for a peacetime tactical wheeled vehicle program will take about the same amount of time to develop, and there is little risk of early obsolescence regardless of which design approach is selected.

For other types of systems, decision-makers should care intensely about the trade-off between capability and cycle time. For example, missile countermeasures for aircraft are typically designed to counter specific threat systems. There is little point to a countermeasure development program that is unlikely to produce any deployable system until after the anticipated service life of the systems it counters. In addition, current capability

gaps in countermeasures are exploitable today by potential foes. Even a partial solution today (or at least soon) would be more valuable than a perfect solution 10 years from now.

It seems only prudent that military planners should have some idea of the time urgency of a given system's development and should take that urgency into account when choosing among proposed alternative system designs. Of course, this would also require both honest and credible schedule estimates for all of the candidate alternatives.

What About Maintaining Technological Superiority?

Much of the concern about acquisition cycle time that has been expressed by DoD officials and Congress has to do with maintaining the historical technological advantage of the United States in military systems. Our all-volunteer force relies operationally on capability overmatch in lieu of sheer numbers, even as it relies morally on exceptional levels of force protection and defensive capability. Given the pace of advance for electronics and cybernetic systems in the private sector and by foreign militaries, staying ahead of the curve would seem to require introducing new technologies on impossibly fast timelines.

As we have discussed above, there are a limited number of potential ways to develop new capabilities on sufficiently responsive timelines. For all but the smallest systems, these methods depend on having taken early and effective action, both within and outside the formal "acquisition" process, in order to be ready to acquire useful systems quickly when the time comes. The two most effective ways to get leap-ahead capabilities on short timelines are both cases of technology insertion. In the first, a weapon system developed as part of a requirements-free technology demonstration project forms the basis for an acquisition program that finishes making the system sufficiently safe, effective, and suitable for operational use. In the second, a novel technology—itself possibly derived from a Science and Technology program or demonstration project—is inserted onto an existing platform or system. In both cases, it is vital that preparatory actions (and spending) have been made in the past—actions that permit the new program to avoid design false starts, inefficient concurrent development of platform and modules, and premature convergence on suboptimal designs. It would seem, then, that the key to being able to maintain technological superiority is to have executed the right set of deliberate actions in the past, on a rolling horizon.

Conclusions

Why Haven't Cycle Times Been Increasing?

In hindsight, it is surprising that cycle times have not shown a general increase over time. We know that the systems we develop and field have grown enormously in complexity, and that this growth is reflected by increased development cost and unit procurement cost, relative to the legacy systems that we replace or upgrade. This is not simply inflation or price hikes; the new systems are much more technologically advanced than the old ones, even relative to the current state of the art in commercial systems. How then have cycle times managed to remain stable?

One plausible answer is that our design and manufacturing capabilities have roughly kept pace with the increased complexity of the systems we procure. This is in part a tautology, enforced by the fact that we cannot build anything we do not know how to build. Those systems that sought to surpass the current state of the art by too much were cancelled, and so do not contribute to the observed statistics.

This is not a stable state of affairs. In particular, software productivity has historically grown much less rapidly than hardware system capability, and much less rapidly than the

amount of software required for our most complex defense systems. In addition, complex software shows diseconomies of scale, so that development efficiency decreases with the size of the software to be developed. As we shift more and more of the functionality of our defense systems into software, we can expect to see a corresponding increase in development lead times.

Finally, we note that these findings may have significant implications for Acquisition Reform. If there are fundamental technical limits to how quickly certain types of systems can be acquired, no amount of management savvy or relief from regulatory burden will allow us to acquire those kinds of systems any faster than that. Reducing cycle time would thus need to involve a combination of using the "ways to acquire things quickly" (enumerated in the What Can Be Had Quickly section) with processes and regulations designed to facilitate those acquisition paths. It would also involve effective oversight to recognize what is feasible as early in the acquisition cycle as possible, and to choose acquisition alternatives accordingly.

Summary

Looking back at the past 30 years of major defense acquisition programs (MDAPs), we note the following patterns:

- The median cycle time and the distribution of cycle times for the majority of MDAPs have been fairly constant.
- The number of extreme outliers from this distribution has been growing over time. Outliers are more frequent in the most expensive programs and in software-intensive programs.
- Cycle time growth has been increasing, especially in C3I and Space programs. Much of this growth seems to be associated with overly optimistic schedule estimates.
- The amount of software in the most software-intensive MDAPs has increased by at least two orders of magnitude over the period in question. There is reason to believe that software (including software-hardware integration) is becoming, or has become, a schedule-limiting factor for these programs.

If program complexity in general, and software content in particular, are now limiting factors for the development lead times of new systems, this has important implications for how we choose which new programs to begin. When it is important to get new things quickly, we will need to test the system designs that are proposed (using credible schedule estimates for those designs) against our best estimates of the urgency and useful life span of the capability being acquired. In particular, we need to be aware when we are asking for an amount of software that cannot plausibly be developed in the time available.

In some cases, we will be able to get useful systems quickly enough simply by asking for less initial capability. For those capability gaps where existing technologies are not sufficient, it would be prudent to invest (on an ongoing basis) sufficient resources in broad technology development and maturation efforts to "keep the shelves stocked" with mature technologies. In parallel with those efforts, we would also need to design and field (also on an ongoing basis) flexible platforms that can quickly incorporate whichever of those as-yet-unidentified future technologies turn out to be needed. The technology development half of that plan is cost-prohibitive if we try to do it primarily within MDAPs; the future insertion half will not succeed if we allow requirements creep during initial development to trade away flexibility for immediate utility.

References

Army Strong: Equipped, Trained and Ready: Final Report of the 2010 Army Acquisition Review (2011). [Decker-Wagner report]. Retrieved from http://www.rdecom.army.mil/EDCG%20Telecoms/Final%20Report_Army%20Acq%20Review.pdf

Brooks, F. P. (1975). *The mythical man-month: essays on software engineering.* Reading, MA: Addison-Wesley.

Jolly, S. (2009a). Is software broken? *ASK Magazine*, 22–25. Retrieved from http://appel.nasa.gov/2009/03/01/is-software-broken/

Jolly, S. (2009b, June). *System of systems in space exploration: Is software broken?* Paper presented at the IEEE Fourth International Conference on System of Systems Engineering, Albuquerque, NM.

Kendall, F. (2014). *Better Buying Power 3.0* (White paper). Washington, DC: Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics

Patel, P., & Fischerkeller, M. (2013). *Prepare to be wrong: Assessing and designing for adaptability, flexibility, and responsiveness* (IDA Paper P-5005). Alexandria, VA: Institute for Defense Analyses.

Putnam, L. H. (1978). A general empirical solution to the macro software sizing and estimating problem. *IEEE Transactions on Software Engineering, SE-4(4),* 345–361.

Riposo, J., McKernan, M., & Kaihoi, C. (2014). *Prolonged cycle times and schedule growth in defense acquisition: A literature review* (Report RR-455). Santa Monica, CA: RAND. Retrieved from http://www.rand.org/pubs/research_reports/RR455.html

Schultz, B. (2014). Please reduce cycle time. *Defense AT&L Magazine, XLIII(6),* 4–7.

Shortening the Defense Acquisition Cycle. (2015). House Committee on Armed Services. Retrieved from https://armedservices.house.gov/legislation/hearings/shortening-defense-acquisition-cycle

Acknowledgments

This work was performed under contract HQ0034-14-D-0001 for the Director, Acquisition Resources and Analysis, OUSD/AT&L.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CA 93943

www.acquisitionresearch.net