Technical Report

CMU/SEI-90-TR-12
ESD-TR-90-213

National Software Capacity: Near-Term Study

Jane A. L. Siegel
Shelby Stewman
Suresh Konda
Patrick D. Larkey
W. Gary Wagner

May 1990

Technical Report

CMU/SEI-90-TR-12
ESD-TR-90-213
May 1990

National Software Capacity: Near-Term Study

Study Co-Directors:

Jane A. L. Siegel
Software Engineering Institute

Shelby Stewman
School of Urban & Public Affairs

Study Staff:

Suresh Konda
Purdue University

Patrick D. Larkey
School of Urban and Public Affairs

W. Gary Wagner
Consultant

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Preface

In conjunction with the Air Force Broad Area Review (BAR), General Bernard Randolph, Commander, Air Force Systems Command, asked the Software Engineering Institute (SEI) and MITRE to perform a near-term study assessing the nation's capacity to produce software for the Department of Defense (DoD). The SEI was also asked to develop a model and methodology to use on a continuing basis to test the health and future capacity of the nation's software industry.

The near-term study began in June 1989, and was managed by the Electronic Systems Division (ESD), Department of the Air Force. Four major tasks were undertaken:

1. Analyses of two major components of the DoD software community:
   - The characteristics of major projects, e.g., application domain, size (source lines of code [SLOC]), personnel requirements of the Air Force, the Army, and the Navy.
   - The characteristics of DoD contractors and subcontractors, e.g., estimated versus actual personnel working on current projects and the previous experience of personnel in development and production of related systems.
2. Analyses of the non-DoD federal government and commercial sectors to enable assessment of the overall labor market supply and demand for national software engineering.
3. Analyses of labor markets for the production of software and of the careers of the individuals involved.
4. Analysis of the supply of labor (U.S. citizens) for the production of software over time.

Primary data sources used to prepare the near-term study report include: self-report questionnaire responses from defense contractor executives and senior Air Force officers; interview data from Air Force, Army, and Navy officials, corporate visits, employment agency heads, and SEI resident affiliates; a National Science Foundation public-use sample on experienced scientists and engineers; corporate proprietary data; and MITRE metrics data.

Numerous secondary sources of data were also used, e.g., Office of Personnel Management reports evaluating the Navy Pay Demonstration Project, General Accounting Office (GAO) reports, the Millburn study of recruitment and retention of DoD scientists and engineers, and Inspector General studies. A complete list of data sources appears in Appendix A.

This document presents the results of the near-term study.

National Software Capacity Study

Abstract: This study provides an initial assessment of the U.S.'s industrial capacity to produce MCCR software. A survey of senior government and industry people showed that 90 percent of them expected a serious problem with the nation's capacity to produce military software over the next 5 years. They ranked acquisition and labor factors as contributing most to the failure of military system development contracts to meet schedule or costs. The study team also analyzed available data about the supply of labor (new graduates and experienced scientists and engineers) and three aspects of demand (Ada systems, PDSS, and related commercial applications) before concluding there is a serious capacity problem. The report describes labor, organizational, and technological issues affecting software production capacity and concludes with some preliminary recommendations for DoD and industry initiatives.

1. The Nation Has a Software Capacity Problem

Our assessment is that the United States has a serious software capacity problem that may worsen substantially unless action is taken on several fronts.[1] This report provides an initial assessment of the nation's capacity to produce and maintain military software, with a focus on mission-critical software. National capacity is dependent upon and affected by other software development and PDSS that is occurring in the non-DoD commercial and government sectors.

1.1. Assessment of Software Capacity by Senior Executives

<u>A survey of senior executives in corporations and government indicates that the nation will have a serious problem in being able to produce mission-critical software over the next five years</u> (see Table 1-1). Respondents included 90 industry and 16 government executives. A high degree of consensus is evident, with almost 90% of both corporate and government executives indicating that they think there will be a problem with the nation's capacity to produce military software over the next five years. Moreover, of those who expect a problem, the severity of the problem was ranked at 4 on a scale of 1 to 5, where 3 = serious and 5 = very serious. Both the degree of consensus and the level of criticality indicate that the United States is facing a serious software capacity problem.

---

[1] This assertion is based on examination of four types of data: a survey from senior executives in corporations and government; data on the demand for software systems, including development and post-deployment software support (PDSS); data on labor supply, both of new graduates and experienced personnel; and data indicating that, given present trends, productivity and labor may fall short of demand.

Table 1-1: Executives' Assessment of National Software Capacity

Percentage of executives who think that there will be a problem with the nation's capacity to produce military software over the next five years.

	Expect a problem	Expect no problem	(N)
Industry Executives	88%	12%	90
Government Executives	87%	13%	16
All Executives	88%	12%	106

The assessed degree of severity of the problem for those who expect a problem.*

	Mean Score	(N)
Industry Executives	3.9	80
Government Executives	4.4	14
All Executives	4.0	94

*Scale:	Very Serious		Serious		Not Serious
	5	4	3	2	1

1.2. DoD Demand for MCCR Software

It is easy to talk in very general terms about capacity, demand, and supply as they relate to software systems and the people and organizations that produce these systems. It is considerably harder to define these terms with sufficient rigor to proceed with a careful empirical study. The central problem is that there are no natural, available measures for the commodity—software systems—that is being supplied and demanded. Ideally, there would be a measure for software of a unit of demand or a unit of supply that could be used in a market analysis in the same way that economists use barrels of oil or tons of steel.

The only measures of demand largely available at the level of projects are dollars (budgeted and actual), person-years, and source lines of code (SLOC). All three measures are fraught with conceptual problems. Dollars and person-years are measures of input units rather than output units. They describe the resources needed to acquire something rather than the something itself. While these units of input may have some systematic relationship with units of output, this relationship has not, to our knowledge, been researched in the software area. Given the difficulty in characterizing units of software output, the absence of empirical studies is not surprising.

SLOC is at least a measure of output. There are, however, significant conceptual problems with the measure for the uses we have in mind. There are differences in the methods for counting lines of code within a single language that make an order of magnitude difference in the result. [Firesmith 88, Hefley 88] The software systems for the ATF (Advanced Tactical Fighter), for example, have been officially estimated to range from 5,000,000 SLOC to 7,000,000 SLOC. [Ada 89, Hughes 88] The variance in SLOC estimates across projects is substantial; metrics tend to be project specific and are therefore difficult to aggregate. There are profound differences among the languages used to write mission-critical computer resources (MCCR) software—Ada, Jovial, Atlas, Lisp, FORTRAN, C, CMS-2, and 78 others—that make the exercise of summing across languages to view total demand a very difficult enterprise, regardless of how consistently the counts within particular languages are done.

Even with good line counts across languages, there are substantial differences in the difficulty of creating segments of code of equivalent size. For example, it is likely that unprecedented 20,000 SLOC systems are much more difficult to develop than precedented 100,000 SLOC systems.

A further difficulty in the near-term effort is that data for even the simplest metrics such as SLOC are not readily available across all systems or even a representative sample of MCCR systems, including MCCR systems in service. In some cases the metrics data are classified. In other cases no one collected and recorded the necessary counts. We have had to work, therefore, with very incomplete, surrogate data on software demand.

1.2.1. Increasing Complexity of MCCR Software

There is general agreement among those close to the acquisition process that new DoD software projects have been increasing steadily in their size, scope, and complexity for about 20 years, with dramatic increases over the past 5 years. This agreement tends to be grounded in anecdotal evidence, crude measures of size (e.g., SLOC or on-board memory), or direct exposure to a few projects over time. There are limited systematic analyses of project size, scope, and complexity across software-intensive projects over time.[2]

To some extent, the belief that software project complexity is increasing is based on a form of backward reasoning; software projects are increasingly over budget, over schedule, and short in performance (relative to the capabilities envisioned for systems); therefore, these difficulties must be because of the increasing size and complexity of systems requirements. Currently, there is no way to determine the extent to which budget and schedule problems are because of increasing size and complexity of system requirements rather than difficulties in the processes of contracting for and managing the development of systems. The data do not even exist to determine how budget and schedule problems are changing over time. While the reasoning about complexity may be roughly correct, it de-emphasizes the role of our nation’s capacity to produce software.

Though unsystematic, the current evidence for increasing project size and complexity is persuasive. To be convinced, one need only compare the software on the F-4 of the Vietnam

---

[2] While there are sporadic claims that simple measures of size correlate well with other indicators of complexity (e.g., overruns on budget and schedule slippage), the claims are inevitably circular; they address the complexity of the solution, not the complexity of the problem. A major difficulty with current complexity measures is that they confound problem complexity with solution complexity; a simple problem inelegantly solved will look as complex as a difficult problem elegantly solved.

era with that on the several generations of the F-16, or with the systems planned for the ATF. The same increases in the size and complexity of on-board software systems can be seen in virtually every category of MCCR systems. Software is increasingly important to the success of various MCCR systems; it has assumed functions that, if they previously existed at all, were hard-wired. Software is an increasingly important component of systems acquisition costs; these increases are particularly dramatic if the calculations include realistic estimates of full life-cycle costs, including post-deployment software support (PDSS).

1.2.2. Causes of Increasing Size and Complexity

The greatest factor contributing to increased project size and complexity is simply imagination. Available or "soon-to-be-available" technologies suggest substantially unprecedented systems that would, if realized, provide orders-of-magnitude improvements in our preparedness. However, our ability to conceive complex MCCR systems has far outstripped our ability to produce these systems. We can easily conceive of MCCR systems capable of sensing and responding to everything in their electromagnetic environment. We can easily see new potentials in standards for integration, security, and fault tolerance. We have had considerably less success in organizing and staffing to realize these ambitions.

There are many other factors that contribute to the increased size and complexity of MCCR systems. We will mention briefly only a few.

Projects are purportedly controlled primarily by "hardware people,"[3] many of whom are not well-versed in software engineering concepts and techniques. The personnel responsible for designing and implementing the software components of systems are often not an integral part of the design process for a system; by the time they are involved, their task is greatly complicated by a host of design decisions made by hardware people. In many instances, the software people talk to customers only through the hardware people. When systems engineering is weak, software development begins late in the project life cycle and must compensate for architectural and hardware inadequacies. Hardware dominance becomes a more serious problem as hardware becomes less important in the overall system. Recent efforts by ATF and LHX acquisition officials led to prime contractors simultaneously assigning top software and systems engineering managers with comparable authority soon after contract award. The impact of this change needs to be documented.

Requirements change and grow over the life of projects as customers and developers gradually come to understand their problem fully. Some of this evolution is driven by unstable perceptions of the threat environment. But much of the evolution is driven by problems in which the hardest thing is understanding what the problems are, not solving them once they are understood. Software problems are complex because they are ambiguous and do not decompose well. Progress on the problems is inherently non-linear and iterative. Often, the problems for ambitious systems are never solved completely and only solved to a tolerable extent well after deployment.

---

[3] The term "hardware people" refers to those responsible for aspects of the system other than software, e.g., aeronautical, structural, and electrical engineers.

Digital hardware technology has changed rapidly in the last two decades. Many of the changes, such as significant increases in speed and capacity for a given on-board weight and volume, are germane to real-time embedded systems. For systems projects lasting 10 to 12 years, the changes in available hardware and software technologies introduce project complexities in at least two ways. First, it is very difficult to commit comfortably to a level of technology and proceed to build a system. New technologies inevitably make early commitments look premature and suboptimal. Changes in hardware technology contribute "moving targets" for software developers that enormously complicate the development process. Second, changes in hardware technology drive changes in systems requirements. Technological advances suggest new functionalities that could be added to systems and new standards of performance for current functions. Given all the changes in hardware, large systems projects need to capture systems architecture and plan for iterative development.

1.2.3. Development Demand

1.2.3.1. Demand for Ada Software

In this section we examine the demand for software written in Ada. Figure 1-1 illustrates the estimated range in current demand for Ada code. The major source of variance in the low and high estimates is omission or inclusion of SDI initiatives that are estimated to require 10 million SLOC. Table 1-2 shows current estimates of SLOC for various military and civilian customers categorized by the life-cycle stage of systems. The high and low total SLOC in Figure 1-1 and data for all categories in Table 1-2 are gross underestimates, especially for systems not yet at the full-scale development (FSD) stage. This means, in turn, that the future PDSS load is dramatically underestimated. While these estimates are only for Ada code, and an underestimate of that, even crude calculations reveal the extent of the capacity problem.[4]

Table 1-3 shows the person-years implied to produce this code. While the original coding demands at the PDSS stage may be much less than at the other stages, error fixes and enhancements are at least as hard, if not harder, than the coding in development. A metric frequently used to estimate software development productivity for both Ada and non-Ada development efforts indicates that an accomplished programmer can produce about 150 lines of integrated, tested, and documented code per month on "easy" (precedented) tasks. This rate drops to 70 lines per month for moderately complex software and 30 lines per month for complex software. [Coles 85] Recent reports of Ada productivity [Myers 88] indicate that average productivity is 77 SLOC per person-month.[5] Even with the most conservative assumptions and all of the missing data, the implied person-years far exceed the nation's current capacity to produce Ada code.

---

[4] For discussions on software labor supply, see Sections 1.3 and 1.4, Chapter 2, and Subsection 3.1.3.

[5] There are reports of specific Ada projects with programmer productivity rates of 550 lines of code per person-month for producing execution environments, i.e., operating systems, communications support, and interfaces.

Figure 1-1: Estimated Demand for Ada Code (Snapshot as of 9/89)

1.2.3.2. Status of MCCR Software

The information in this section is based on an examination of MCCR projects from two Air Force commands for five application domains: avionics, communications, command, control, communications and intelligence (C^3I); electronic warfare (EW); and radar systems. Software size, schedule, project management and skill base, and budgetary data are the key indicators of project status we analyzed. For the near-term study, we concentrated on the Air Force and were able to access data on some of these indicators for 37 projects.[6]

Software Size

The range in amount of software in terms of SLOC varied from a low of 4,000 lines of code to 6,000,000 lines of code. [PropDat1 88] [PropDat2 89] Avionics projects studied involve much larger amounts of software on average than did any of the other application domains. For communications, C^3I, EW, and radar systems taken together, the average SLOC is 260,000, with a range of 4,000 to 520,000 SLOC. [PropDat1 88]

---

[6] Data constraints for the near-term effort included: (1) insufficient time to make required nondisclosure arrangements for some or all of the data and to collect and analyze data; (2) difficulties in locating time-series data; and (3) lack of availability of data on the software part of overall project budgets.

Table 1-2: Estimated Number of Lines of Ada Code Planned, in Full-Scale Development and PDSS Stages

The growth of software size during the software and systems development for projects was difficult to document. Among the 37 projects studied, there were 3 cases where we had access to documentation indicating that the actual amount of software increased by more than 100% of the estimate made at the time of contract award. The frequency and magnitude of change in software size for MCCR projects need to be measured consistently, since these changes seem to contribute in a major way to schedule slippage and cost overruns.

Schedule Slippage

The average schedule slippage for reaching the Formal Qualification Testing (FQT) was 18 months[7] for a sample of 35 projects from among 101 MCCR efforts funded by Electronic Systems Division (ESD) in 1988. One project reached FQT as scheduled, and three slipped by more than 30 months. Factors associated with the worst cases of slippage included: insufficient skill base; contract provisions (especially reaching the ceiling on fixed-price con-

---

[7] We used estimated time from contract award to actual or updated estimates of FQT from proprietary sources to make this determination. Data for both points in time were available from 17 projects studied.

Table 1-3: Labor Equivalence Figures in Person-Years for Easy, Moderate, and Difficult Code Based on the Low Estimate of Lines of Ada at Each Stage of Development tracts that removed contractor motivation to complete work in a timely fashion); and near litigation over requirements changes that shifted the focus from experienced technical personnel trying to produce a product to business personnel posturing for financial and administrative purposes. [PropDat1 88] [PropDat3 89]

Project Management and Skill Base

Technical project management by contractors was noted as a serious problem in 9 out of 29 projects. [PropDat1 88] Similarly, skill-base inadequacies were a problem in 16 out of 29 projects. There were four cases where the need for more personnel with application domain experience was documented, and four cases where concern over contractors shifting personnel to other contracts was noted.

Estimates of the skill base for software development projects appear to be determined in many different ways. [PropDat1 88] At the time of the contract award, the number of person-hours estimated to do the software development on projects is often calculated on the basis of estimated software size. Given actual versus estimated software size differences, we can expect substantial shifts from estimated to actual labor requirements, and a concomitant increase in project cost.[8]

Budget Overruns

To estimate the cost to the nation of developing MCCR software systems, we need information from contract award through system delivery on the estimated and actual experience for both the overall system and its software. It seems difficult to get realistic cost estimates for the software portion of some MCCR projects. For 3 of the 37 projects we studied, the software portion varied from about 20% to about 38% of the total cost of the system. Getting data over time and gaining information about recently completed systems so that actual costs may be determined will be a key task of any long-term national capacity effort. At present, data on cost overruns of MCCR systems for the software part alone tend to be well documented only when there is litigation or audit activity. Of the 37 projects, 3 projects where data was available for the near term had overruns of about 100%; i.e., for a project originally estimated at $15,000,000, the cost estimate for completion is now about $30,000,000. [PropDat1 88] [PropDat2 89]

Overall Status

To get an overall sense of the status of MCCR projects, we looked at the green, yellow, and red designations for 1988-89 for 26 communications, C^3I, EW, and radar systems projects. Overall, 6 projects were rated red, 9 were rated yellow, and 11 were rated green. These data suggest that 58% of current projects are experiencing difficulty. Difficulty means these

---

[8] Data on skill base are available, but getting data where comparable estimation techniques are used for the population of projects within an application domain or a government command or service is difficult. These are the data needed to do national capacity estimation.

projects are likely to deliver systems to the government at least one and a half years later than estimated at contract award time and will experience substantial skill-base shortfalls and cost overruns.

1.2.4. Post-Deployment Software Support (PDSS)

Perhaps the most rapidly growing segment of the military service software workload is in PDSS. Each step in the evolution of the inventory of operational MCCR systems increases the PDSS load; there is more software, and it is more sophisticated.

Table 1-4 shows very conservative estimates of the growth in PDSS costs and personnel requirements from FY87 to FY92. Table 1-5 breaks this data out by year. The largest increases in both personnel and dollars occurred from 1988 to 1989. The growth is impressive even though it is almost certainly understated.

While software does not degrade with time in the same way as hardware, PDSS activities are managerially and technically formidable. The work is a mixture of fixing errors and enhancing the product. The work usually involves, at the outset, creating more useful documentation to replace the large amount of documentation generated to satisfy the 2167A standard, because 2167A does not require some critical information. The new documentation must, for example, capture the design rationale and architecture of the system if a succession of professionals is to work effectively on fixes and enhancements. A substantial amount of the initial work in PDSS is, in fact, working with the development contractors to bring systems to performance levels that were required but not reached prior to acceptance, and in some cases, deployment.

We did not have access to studies of the impact of software on operational readiness. We hypothesize that software is an increasingly important reason for operational failures of all kinds. There is certainly a rich set of anecdotes about systems failures, including catastrophic failures, caused by software failures.

PDSS activities also pose another important labor problem. PDSS is in many instances at least as difficult technically as original development. Yet the PDSS activities often are held in much lower regard than original design and development activities among software professionals. This low status makes it very difficult to attract and retain the caliber of professionals needed for these critical activities. Many people do not want to devote their careers to what is essentially cleaning up the messes made by others.

The Air Force has taken fairly drastic steps to cope with the increased PDSS workload. These steps may not be adequate. For reasons discussed below, the military services have extreme difficulty in acquiring and retaining the level of software talent required for in-house PDSS. There are also serious problems with continuing reliance on contractors for PDSS. To be more concrete, it is not clear, for example, who would maintain the software on Army MCCR systems in the event of a war or other circumstances requiring civilian evacuation in Europe.

Table 1-4: DoD Summary of PDSS Costs of Embedded Systems and Test, Measurement, and Diagnostic Equipment, Broken Out by Service

Table 1-5: Organic Versus Contractor PDSS Personnel Costs for Embedded Systems and Test, Measurement, and Diagnostic Equipment for FY87-FY92

1.2.5. Growth in Non-Defense Demand for Comparable Software

Until fairly recently, the military has had a virtual monopoly on a large class of computing applications and has been the predominant employer of the practitioners with the interest and the skills to solve such problems. Historically, there were only a few civilian enterprises, most notably communications, computing, and education, that competed directly with the DoD for scarce personnel in science/computing and engineering/computing. This circumstance is changing rapidly. There has been substantial growth in the civilian communications and computing industries and in new applications in large industries such as health, transportation, and manufacturing. While most civilian applications continue to lack the time-critical feature of MCCR systems applications, there is a proliferation of applications requiring sensing and real-time software for acquiring, interpreting, and presenting data.

Another burgeoning area of roughly equivalent applications is in sophisticated process controls in manufacturing. These controls represent one of the few areas of potential comparative advantage for U.S. industry in its struggle to remain competitive. The extensive efforts of the Japanese and Europeans on manufacturing process controls may deny any U.S. advantage, in spite of the current superiority of U.S. digital technology and software engineering. The problem is familiar. The critical applications employ the last generation of technology. Over the past 30 years, the U.S. advantage in frontier technologies has not translated well into an advantage in applying known technologies to practical industrial problems. For the concerns of this study, the main point is that non-DoD industry in the U.S. will make even larger efforts on process controls and will increasingly draw on the same personnel pool as the DoD in its MCCR activities.[9]

A few civilian agencies of the U.S. federal government are also increasingly competitors for personnel with MCCR-relevant skills. The three most important agencies, because of their size and relevant applications, are the Federal Aviation Administration (FAA), the National Aeronautics and Space Administration (NASA), and the Department of Energy (DOE). The software system for the new Air Control System for the FAA has been estimated to be as large (10,000,000 SLOC) as the software in all but the most ambitious MCCR systems. The system is real-time, similar to many military systems in its physical application domains, written in Ada, and subject to stringent time-critical performance standards. The system has competed and will continue to compete directly with DoD for real-time MCCR talent.

While we have been unable in this brief near-term study to quantify and forecast the civilian demand for the scientific/computing and engineering/computing skills critical to the development and PDSS of military systems, the qualitative evidence clearly indicates that the DoD monopoly on a large class of computing applications is ended. At best, the DoD must pay a substantial premium for the skills it requires. At worst, the DoD will find the requisite skills unavailable at almost any price.

---

[9] Evidence that the pool is a national one with DoD and non-DoD competition is shown in Chapter 2; see Section 2.3.2.

1.3. Changes in the Supply of Technically Qualified Labor

The changes in the supply of technically qualified labor exacerbate the capacity problem. There are several elements of the problem that may counteract any gains that might have been expected from decades of increasing budgets for education and scientific research in the United States. This section examines changes in the supply of professionals who are expected to produce the bulk of MCCR systems in the future by describing recent trends in the educational pipeline and characteristics of labor mobility.

<u>Enrollment in engineering and science programs is not increasing rapidly or is experiencing absolute declines.</u> From 1976 to 1986, the number of baccalaureate degrees awarded per year in the sciences declined from 253,000 to 247,000. After a rapid increase from 1976 to 1983, engineering baccalaureate awards remained stable at roughly 77,000 per year. During the same period, science and engineering master's and doctoral degrees increased modestly from 54,700 to 62,500 and from 17,400 to 19,200, respectively. [NSF 88] Universities and colleges expect a continuing decline in science and engineering enrollment as total enrollment declines, with relatively fixed proportions of students enrolling in science and engineering programs. Even in the computer sciences, an area that had displayed rapid growth during the first half of the decade, [NSF 88] enrollment at the undergraduate level has declined and the number of new PhD students appears to be dropping. [Gries 87] Degrees granted in undergraduate computer science and computer engineering programs remained approximately stable from 1987-88 to 1988-89 (10,759 versus 10,688). The master's level also remained stable during the same period. [Gries 89]

While more women are entering the engineering and scientific professions, these professions continue to lag behind other professions. The initial increases in female participation in university engineering programs from the 1% to 2% levels of 25 years ago could level off near the current 15% level unless there is continued widespread encouragement from employers, the educational community, and society at large. Nor can we reasonably expect that pressures on the science and engineering labor force will be eased through increases in the numbers of black and Hispanic scientists and engineers. These groups have not entered the science and engineering labor force in significant numbers in the past; given current trends, they are not expected to do so in the future. [NRC 86]

From 1976 to 1986, the number of employed scientists and engineers grew at an annual rate of 6.3% (8.6% for scientists and 5.9% for engineers). Women employed in science and engineering accounted for 8.2% of the work force in 1976 but 13.4% by 1986. For engineering alone, the comparable figures are 1.6% and 4.1%. This represents better than a 16% annual rate of growth but still leaves women accounting for only 92,600 of the 2.2 million employed engineers. In comparison, the growth of black and Hispanic computing engineers is encouraging, but the base levels are so low that it will be many years before any pressures on the labor force can be relieved by these groups. [NSF 88]

Graduate enrollments in engineering and science programs also show increasing representation by foreign students. In the fall of 1983, over one third (34.3%) of all engineering graduate students were foreign. At doctorate granting institutions in the U.S., the percentage of foreign graduate students has increased between 1976 and 1983 from 34% to 42% in engineering and from 24% to 38% in computer science. As for doctorates awarded, in 1977 43% of all engineering and 14% of all computer science doctorates were awarded to foreign students (see Figure 1-2). These percentages changed by 1983 to 56% of all engineering and 36% of all computer science doctorates. [NSF 85][10] In 1987-88, the proportion was 41% for computer science doctorates. [Gries 89]

Figure 1-2: Percentage of Doctorates Awarded to Foreign Students

In addition, the relative Graduate Record Examination (GRE) verbal and quantitative scores by U.S. and non-U.S. students (1980-81) are troubling. Of those enrolled in graduate study, the average verbal scores were 462 for U.S. students and 360 for non-U.S. students. The 22% differential, as might be expected, favors U.S. students, since the language of the test was English. However, for the quantitative scores, the U.S. students scored 471 and the non-U.S. students 569; the score by U.S. students is 17% lower. [NSF 85] These figures indicate that the U.S. students coming through the education pipeline are in trouble. While those who opt for graduate study in mathematics or science are about even in quantitative scores, the remaining U.S. graduate populations have lower scores. In sum, the current populations of U.S. graduate students represent a shrinking share of those trained in science, engineering, and computer science; among all graduate students, the U.S. students are notably less well trained in mathematics; and the forthcoming demographic shifts leading to smaller U.S. student populations all speak to a serious current problem and an even more serious long-term pipeline problem.

Because foreign nationals cannot get high-level clearances to work on DoD projects, they

---

[10] In the physical sciences, this figure was 24%.

move to the commercial sector, or academia, if they work in the U.S. at all. A separate issue is that there is a substantial offshore work force of Asian programmers and systems experts organized around programming "sweatshops." [Singhal 90] Because they work for below market wages, they allow software development costs in the commercial sector to be reduced or salary savings to be transferred to other employees. The impact of this factor, when combined with the clearance issue for foreign nationals, is to provide relatively greater increases in the supply of computer professionals to the commercial sector at the expense of the defense industry.

There is also some evidence that DoD contractors will sacrifice their demands for needed skill levels to hire software personnel who have security clearances. The tradeoff is made because the cost of hiring people without clearances can be enormous. It was reported that it is not uncommon for engineers with critical skills to be hired and then put "on ice" for as much as a year before they receive clearance to perform the jobs for which they were hired. Engineers with needed skills are apparently in such short supply that employers are willing to bear the cost of an idle employee’s salary to be sure that personnel with critical skills are available when they are needed.

This picture is further complicated by the flow into and out of engineering and computing specialties. About one-sixth of the 1984 workforce holding engineering jobs had degrees in fields other than engineering, and about 80% of the computer specialists had degrees in other fields. Finally, more than one-third of those with engineering degrees were employed in non-engineering occupations. [NRC 86]

Industry contractors and DoD civil service agencies confirm that many of the employees who design and generate software for military systems hold degrees in fields other than computer science or management information systems. In fact, employers often express preferences for hiring people whose primary expertise is in specific application areas such as radar or optics. In these cases, their software skills are considered to be secondary to their engineering or physical science expertise. One consequence of these hiring preferences is that training for the software professionals who are to generate MCCR systems will continue to take place on the job. Either physicists and engineers will hone their specific applications skills while learning the newest software engineering techniques, or software experts will gain sufficient engineering and physical sciences training that they can contribute more than efficient code to projects. In either case, training on the job will be a lengthy process, and attempts to greatly alter the labor supply in the short run are unlikely to be successful.

The rapid expansion in the number of degree holders in information systems and computer science may do little to relieve pressures on the supply of software engineers needed for MCCR systems, so long as this situation obtains. Nevertheless, it is important to note that Bureau of Labor Statistics projections indicate substantial increases in the supply of computer programmers and computer systems analysts over the period from 1986 to 2000. Estimates vary from about 50% to 75% over the period, but under either set of predictions, supply is expected to lag behind demand. How these increases are accomplished, whether through career changes from the sciences and engineering or from non-scientific fields, will profoundly influence MCCR systems capacity. [White 89]

Given the above assessment of capacity, it is clear that major increases in the total number of software personnel will be required. Our data indicate, however, that it is not just numbers that are relevant; it is also shortages in specific critical skill areas within the software labor force. Equally important is the strong message that the capacity problem cannot be solved by dealing with labor or personnel alone; productivity must also be addressed, particularly with changes in technology and changes in organizational and management policies and practices.

1.4. Labor and Productivity Gains

Different sectors—the military, the U.S. Civil Service, and industry—have different kinds of software skill shortages and different organizational problems in managing software personnel (see Chapter 2). The interaction of personnel among the three sectors is also an important managerial and human resource issue. However, regardless of what portion of work is conducted internally by the DoD and the U.S. Civil Service or contracted out to industry, the numbers of entering software personnel must be increased, whether by new graduates or by those already in the work force but not working in software. A different dimension that might alleviate *part* of the numbers shortfall on entrance involves addressing retention rates of those already in the software field.

Another important dimension of the labor problem is the shortage of specific critical skills. Both in the senior executive survey and in interviews with corporate managerial and software personnel, it was made clear that the software capacity problem is not simply a numbers problem. A critical component of the software labor problem is one of shortages in critical skill areas. Those identified as most important for capacity are systems engineers, application domain experts, software engineers, software managers, and project managers.

According to the senior executives we surveyed, four of the top six factors contributing to the failure to meet schedule or costs on development contracts for military systems were software labor shortages—not in overall numbers, but in critical skill areas. Table 1-6 indicates the views of the senior executives on the relative importance of factors affecting cost and schedule slippage.

The separate corporate interviews revealed that the most acute personnel needs are for systems engineers and application-domain experts; particularly needed are professionals with experience across an entire project and with end-user insight. Similar experience is required of effective software and program managers. Since new graduates obviously do not have such experience, the required skilled personnel will not be produced overnight. It is increasingly important to have a more systematic long-range approach to develop and retain such personnel for all sectors—the military, the U.S. Civil Service, and industry.

Table 1-6: Executives' Perspectives on the Relative Importance of Factors That Contribute to the Failure of Military System Development Contracts to Meet Schedule or Costs

Factors	Mean Score[a]
1. Inadequate requirement specification	4.5
2. Changes in requirements	4.3
3. Shortages of systems engineers	4.2
4. Shortages of software managers	4.1
5. Shortages of qualified project managers	4.1
6. Shortages of software engineers	3.9
7. Fixed-price contracts	3.8
8. Inadequate communication for systems integration	3.8
9. Insufficient experience as a team	3.8
10. Shortages of application domain experts	3.6
11. Integration of contractor/subcontractor efforts	3.5
12. New application domain	3.5
13. Inadequate software development tools	3.4
14. Turnover of personnel	3.3
15. Shortages of skilled programmers	3.2
16. Complex hardware	2.8
17. Shortages of electrical engineers	2.7
N[b]	(105)

Other factors were suggested by 32 respondents as follows:

Subject	Number of Times Mentioned if > 1
Acquisition (contract process)	7
Costs, pricing	6
Process understanding	4
Schedules	4
SPO's	3
Turnover	2
Planning	2
Training	2

[a]

Scale:	Very Important		Important		Not Important
	5	4	3	2	1

[b] The numbers vary per item, but in general include 105 respondents.

If, in addition to development, we consider PDSS, then experienced people—with skills in critical areas that reach across an even larger domain—are needed, again in all sectors. Since the full PDSS load has yet to arrive, there is time to consider its impact within the larger picture. At present, the PDSS load might be viewed as the tip of an iceberg. But in the future, the rest of the "PDSS iceberg" must be accommodated, while at the same time dealing with the increasing load of new projects. Thus, it should come as no surprise that addressing the need for both increased numbers of software personnel and increases in the critical skill areas is necessary, but not sufficient. A more comprehensive approach to increase labor pools and productivity is required.

The importance of a more comprehensive approach was indicated by senior executives in both government and industry; by interviews with military, U.S. Civil Service, and corporate managers and technical staff; and by the gap indicated between the trajectories of the demand for software and the supply of software personnel. For instance, the requirements specification process and changes in requirements (see Table 1-6) were identified by industry and government senior executives as the two most important factors contributing to the failure of military systems development contracts to meet schedule and costs. These factors pertain to macro organizational and managerial issues among different organizations and organizational staffs and to the evolution of product design and invention, which have both organizational and technological components. Interviews with industry executives also indicated that for their own corporations, it was important to increase both the number of software personnel and internal productivity to meet their own corporate demand trajectories—those for the recent past and those expected over the next several years.

Initial efforts to solve the long-range capacity problem by technological jumps in productivity, one of which rests with expectations for Ada, for instance, may also exacerbate the problem in the short run. Use of prior modules in other languages and small modifications on "reuse" of such applications must, at the onset of a wide new Ada initiative, create an increased problem because of discarding of old, but *operational* code, and because of shortages of personnel with expertise in Ada. Also at issue is the extent of Ada use by the rest of the software world—non-DoD government and commercial industry—potentially affecting the exchanges of personnel in the overall labor market and the entrance of new firms working in both the DoD and commercial markets. In brief, all three components—labor, organizations/management, and technology—need to be addressed simultaneously to begin to solve the national capacity problem.

If, in the future, capacity lags yet further behind demand, it will be crucial to stay informed of the gap and to measure its magnitude more accurately. Alternatively, if actions begin to narrow the gap, it will be important to be informed of such changes and plan accordingly. Despite the requirement that national-level data include all three military services (military and civilian support), non-DoD government (e.g., NASA and FAA) and industry (DoD and commercial), there is no overall database currently available to handle the task.

Future efforts should be directed at developing and archiving a national database and at developing a national-level macro model for estimating national capacity over time. The database would be used for national-level estimates and forecasts of the capacity of the nation’s software industry, and for input regarding how changes in labor, organization/management, and technology affect the nation’s capacity to produce software for the DoD.

2. Labor Markets and Human Resource Impacts on Capacity

Wages and salaries have significant behavioral impacts, but they are only one among an array of motivational factors affecting entry, performance, and retention. Conventional labor investigations center on wages and salaries and do not recognize organizational impacts on labor markets, just as in the general economic theory of markets, organizations and organizational decision makers are omitted.[11] More recent work by internal labor market (ILM) theorists, in contrast, places great attention on organizations and neglects the flow of personnel between ILMs within an organization (for example, between occupations), and also neglects the flow of personnel across organizations. This study recognizes the limitations of both approaches and examines not only organizational ILMs, but also national labor markets. Three themes guide the discussion:

1. Career ladders and how they differ across sectors (military, civil service, and industry).
2. Shortages of skill by sector and some of the reasons for these shortages.
3. Two types of labor mobility:
    - The movement of labor into and out of science and engineering, and, in particular, into and out of software work.
    - The movement of scientists and engineers into and out of the DoD industrial sector.

2.1. Career Ladders

2.1.1. Industry

<u>In industry, both MCCR and commercial, the career ladders of technical people might be described as having a **Y** shape.</u> (see Current Industry in Figure 2-1.) At the lower grades, promotions and lateral movement occur in science and engineering jobs, increasing the technical responsibility and scope of the incumbents and initiating team management and supervisory responsibilities. However, after several promotions there is a more clear-cut demarcation: <u>one can move upward into management or remain in a more purely science and engineering track and move upward.</u> The number of slots on the technical side are considerably fewer than those on the managerial side, but industry has created another branch to retain its brightest and best scientists and engineers. In the past, the industrial technical career ladder might have been described as a left-handed **Y** without the upper right branch of the **Y** (see Old Industry in Figure 2-1.)

---

[11] For this latter point on the economic theory of markets, see Cyert and March 1963 [Cyert 63].

Figure 2-1: Software Development Career Ladders

2.1.2. Civil Service

The U.S. Civil Service career ladder for scientists and engineers, in contrast, is a truncated form of the above industrial structure: an abbreviated left-handed Y with the upper part of the right-hand branch lopped off and with the left-hand branch pruned severely. (see Civil Service in Figure 2-1.) There is essentially no parallel upward track for scientists and engineers (the right branch of the **Y**), and movement into the upper government service (GS) levels of management is not only limited in number, but basically capped at GS-14 and GS-15. Most of the bottleneck or career plateau is at GS-12 and GS-13. It is at these locations that after rapid movement upward from, for example, GS-5, GS-7, or GS-9 (depending on the degree at the entry point), high exit rates occur. The best and brightest of those within the civil service have essentially nowhere to go but out. We characterize the civil service *technical* career ladder[12] as basically an **I**, but this **I** is much shorter in height than the industrial **Y**. The **I** part is, in fact, the base or lower stem of the **Y**, and it could be called a dwarf **I**.

2.1.3. Military

The story for the military is similar to the story for the civil service, except that in this set of science and engineering organizational labor markets, rotating job assignments (command -> technical -> command ->, etc.) severely limit the ability of military personnel to get to the forefront of technological expertise and stay there. The technical career ladder could again be described as an abbreviated, left-handed **Y**, as in the civil service.[13] What needs to be noted here, in addition, is that at least part of the trunk or stem of the truncated, left-handed **Y** is missing because of non-technological, command job assignments. This problem becomes especially acute when a technically trained officer's first assignment is to a command

---

[12] For scientists and engineers.

[13] We are not attempting to make comparisons between military rank and GS level.

billet. In this instance, the career ladder is an abbreviated, left-handed **Y**, but with part of the base or trunk missing (see Military in Figure 2-1), with the missing parts representing non-technical, command assignments. In this type of ILM there is no necessary cap on career if the decision to leave the technical field is made early, but for those whose preference is on the technological side, the only option is out. And even before that option is reached, part of the technological edge will have been lost. It would therefore seem that the options for the best and brightest personnel are two:

1. To get out of technical job assignments as quickly as possible, opting early on for the more elongated managerial or command branch, since there is no comparable technical branch.
2. To stay, where possible, in a sequence of technical assignments if these are at the forefront and enable one to gain valuable technological experience and maturity (e.g., application-domain experience), and then to opt out as quickly as possible once this growth curve turns down or a technical job assignment is made that blunts one's skills at the technological edge.

If the latter alternative existed, in fact, then the incumbents would experience a sequence of challenging technological job assignments, allowing mastery and growth. Moreover, if the authorization or allocation of such jobs were sufficient in size so that continuing groups of entering scientists and engineers could be assured of such a technological edge, then the problem would shift to one of continuity of knowledge between groups.[14] However, currently it appears that while sufficient numbers of science and engineering graduates are entering the military, a significant portion are not being honed for work at the *practical, operational* technological edge or frontier.

This observations holds whether the frontier is defined either as:

- Developing new MCCR systems that perform as designed with end users in mind.
- Technological capacity for maintaining and enhancing the operational readiness of the more technologically sophisticated MCCR systems, both those that the U.S. currently has on board and also those that are expected to be on board in the next decade.

In brief, the career ladder options in the military would appear to follow the representation for either Old Industry or Military in Figure 2-1. The first option would be to move out of technical assignments as early as possible; the second option would be to remain in technical assignments and face a truncated career. In the latter case, the career would be capped early for all but the best and brightest, and capped almost exclusively at the Major level for even the best and brightest MCCR personnel.[15]

---

[14] But, there would be a continuous stream of highly motivated junior-level technological experts.

[15] These data pertain to the Air Force, but it appears that a similar problem also exists in the other services.

2.2. Shortages of Skill Types

2.2.1. Industry

As noted in Section 1.4, one of the major factors contributing to the failure of software systems development contracts to meet schedule or costs is a shortage of people with critical skills. The following rank ordering in terms of most acute shortage was indicated in the senior executive survey of Table 1-6: systems engineers, qualified project managers, software managers, software engineers, and application domain experts.

In corporate interviews one critical skill area consistently stood out as having the most acute shortages: systems engineers. Moreover, it was stressed that there are different kinds of systems engineers—specialized by application domain, for example—and thus we may more appropriately speak of shortages of systems engineers in each of the primary application domains. As noted by one corporate manager, "anyone can design a system; the real challenge is to make it work." Furthermore, it is essential that systems engineers have application domain expertise and prior experience working on a project from beginning to end for at least one major project. There is not only a shortage of systems engineers, but rather a shortage of experienced systems engineers.

Layered on top of this shortage is another problem that will take on increasing importance as MCCR projects become more complex: it will take teams of experienced systems engineers to integrate them, and this evolution also means that it will require a new type of project manager or managerial team. This emergent issue highlights the fact that the shortages have both a critical skill component and a team component, different in kind from less complex projects. In this respect, simply solving past shortages will not be sufficient. A more sophisticated approach than has generally been used before is needed, dealing with critical skill areas, and with the interface and interaction of these skill areas at a new level.

Another emerging dimension of the critical skill area shortage for industry is the changing place of software in maintaining a competitive advantage. While in the past the competitive edge was in hardware technology, some of that edge has now shifted from hardware to software technology. Software embodies more of the functionality of the final product. Thus, while software is a one-shot affair in the development to production phase and is not a large profit maker in the relative sense, it is now much more crucially important to the product. Hence, technological finesse is expected to become increasingly associated with software. It is from this perspective that industry is placing considerably more importance on systems engineers and software engineers both in terms of number of jobs *and* in the importance of such work to maintain a technological edge in the marketplace.

Finally, there is intense competition among corporations with major DoD contracts because of the national market for systems engineers and because of the partial segmentation of the DoD share of that market. In such a labor market, a premium is placed on experience over the full life cycle of a project and in specific application domains. The pool of these personnel is quite small. Thus, in these "micro-markets," corporations rely on inter-organizational contact networks to identify key experienced personnel. In addition, they keep an eye on competitors who lose out when contracts are awarded, providing ideal timing and targets for hiring away experienced personnel. While this process may be efficient for allocating personnel in organizations and jobs where they are needed at that time, it does not help alleviate the overall shortage in critical skill areas, since it is a form of musical chairs. The competition from non-DoD corporations for systems engineers may also cause decreases in these critical skills when corporations experience ups and downs in the awards process. Reliable estimates on the exit rate from the DoD industrial sector by critical skill area, including application domain and project experience, are not available, but there are national-level data for estimating exchanges between sectors, which we examine in Section 2.3.2.

2.2.2. Civil Service

In general, there is both a quantity and a quality problem in the civil service sector[16] because of difficulty in recruiting and retaining qualified technical personnel. Except for the Navy Personnel Management Demonstration Project, recruitment of highly skilled software personnel is difficult, especially in geographical areas with high technology industry. Evidence of recruitment problems include the waiting time to fill vacancies, pay differentials between the civil service and private industry, recruitment in general from second-tier universities,[17] and inability to get the best students at the second tier. [PropDat2 89] The extent of difficulty in recruitment varies by geographical region and is dependent on the extent of high technology industry in the region. However, it is clear that at present the civil service is not competitive with industry in either salary or advancement opportunities,[18] both of which affect entry. As indicated below, these same two factors, salary and advancement opportunities, are also major problems in retention.

High turnover is a dominant characteristic of the software industry in general. However, while it is a constant irritant in the commercial sector, it is often a major threat to systems development and PDSS in the DoD and civil service sectors. Since the majority of civil service turnover appears to be voluntary, most people who leave fall into one of two groups:

1. The best and the brightest.
2. Young people who honed their skills and augmented their education in civil service and DoD positions.

Approximately 8% of the scientists and engineers employed by the civil service and working on DoD-sponsored projects can be expected to leave each year. That figure represents the equivalent of a complete turnover of the working force every 12.5 years. Initial estimates suggest slightly higher figures for those whose work centers on computing.

---

[16] Working for the DoD.

[17] By "second tier" we mean those universities not ranked as the top institutions of higher education in the U.S.

[18] For the latter, see Section 2.1 on career ladders.

Responses to these turnover levels by the civil service vary. Revisions in the Civil Service Act allow for increased flexibility in pay and promotion of scientists and engineers, although it is still uncommon to find such employees above the GS-14 level. A review of the Navy Personnel Management Demonstration Project showed that offering salaries above the civil service averages and encouraging greater involvement of supervisors in goal setting and performance evaluation, coupled with a performance-based pay scheme, reduced turnover among scientific personnel even though pay levels at the demonstration bases continued to lag behind those in the private sector. What may be most significant is that retention at the demonstration bases increased noticeably for recent hires and superior performers. [Personnel 88]

In August 1989, Dr. George Millburn, deputy director of the Department of Defense Research and Engineering, reported preliminary findings from a major study on recruitment and retention showing that the largest losses of civilian scientists and engineers from the DoD laboratories continue to be from the most experienced levels. [Millburn 89] Of those leaving, 50% resigned from the federal government, and the two-thirds of those who left to take jobs in industry most frequently cited salary and opportunity for advancement as their reasons for leaving. In 1986, loss rates were higher than in 1981 for every grade level except for GS-5 and GS-7. Furthermore, it takes so long to replace employees who leave that vacancy rates in the laboratories reached the 6% level at the end of 1986; approximately 1,500 of 25,000 positions were unfilled.

In summarizing the factors that influence scientists and engineers to join, stay, or leave DoD civil service employment, the Millburn study noted that these employees join because of geographical factors and the nature of the work; they express satisfaction with their supervisors and meeting organizational objectives; they express dissatisfaction with the way they are treated by the DoD and with their promotion opportunities; and they indicate that better pay and advancement opportunities are the most important factors in their decision to leave.

A comparison of 1987 salary levels for scientists and engineers in DoD laboratories and in industry research and development establishments shows a consistent pattern of lower pay in civil service positions for all educational levels, for non-supervisors, for supervisors (except those in the lowest deciles), and for division directors. The differentials generally increase with pay decile, so that the most valued civil service employees (as measured by their pay) lag the farthest behind their industry counterparts. The best paid non-supervisory scientists and engineers in DoD civil service trail their industrial counterparts by about 14% for bachelor's and master's degrees and by about 18% for PhDs. At the median, the corresponding figures are 8% and 14%. First-level supervisors in the DoD civil service trail those in industry by 10% at the median pay levels and by 18% at the 90th pay percentile. Comparable figures for division directors are 17% and 27%. The best and the brightest have the greatest gap on the factor judged most important in the decision to leave.

Our interviews suggest that although the DoD hires new scientists and engineers predominantly from second-tier universities and colleges, it appears that the DoD will be able to attract its share of recruits. However, we believe that it will continue to have problems retaining them for the foreseeable future. Therefore, the pool of experienced scientists and engineers is likely to be degraded in quality by turnover levels. Information gained from our interviews supports Deputy Director Millburn's conclusions that there are serious skill shortages at the laboratories in artificial intelligence, systems engineering, and computer engineering. Furthermore, it is at the more advanced skill levels, where pay differentials are greatest, that recruitment and retention pose the most serious threats to capacity.

2.2.3. Military

While the military would appear to have a very capable pool of talented entrants in science and engineering, there are two major personnel management problems affecting the military side of the capacity problem. First, identification and further technological development of the best science and engineering officers is seriously inadequate, often resulting in much weaker technical skills than those in industry or the civil service. This has major consequences for the technical interchanges in the management of the acquisition process. It also has strong implications for operational readiness. When something breaks or does not operate as designed, who will fix it or be able to see that it is made operational in the shortest time possible? How much reliance should there be on contracted labor and civil service personnel, especially in wartime or emergency situations? Given the technological sophistication of the MCCR systems now deployed and those coming downstream, how can any less emphasis be given to the technological sophistication of the human resources who bring them online and maintain them? Regardless of the degree to which development and PDSS are a tri-sector cooperative effort involving contracted industry, DoD civil service personnel, and military personnel, the level of managerial and technical skill should be complementary and have a high degree of *comparability* across the three sectors to ensure the most effective teamwork at the macro level. Presently, the military is not developing its technological human resources to most effectively engage and manage that teamwork. The selection processes do not reflect differential abilities, nor do they map the individual's skills and skill sequence at the "micro-market" level, as is done in the industrial sector. Private industry manages such efforts at a much more fine-grained micro level, with the goal of recognizing and honing such talents. For instance, private industry stresses growth in knowledge of application-domain; attempts to assure continuity of assignment to attain a targeted technical experience level and to experience the full range of team building; and attempts to match individual interests to job assignments at sufficient granularity to motivate and differentiate those individuals with distinctive insights or curiosity about the internal technological logic of processes.

Second, and closely related, are the job assignment and career advancement processes that severely handicap the refinement of skills and limit the aspirations of those who would opt for a more technological level of expertise and career line. On the one hand is the discontinuity of technological job assignments, especially of first assignments. The competition in private industry more finely tunes its new entrants over the initial three-year period so as to sharpen their technological skills for *practical, operational* use. In general, there are few comparable types of job assignments in the military for technical personnel, and particularly for those in MCCR software applications. Those who have the technical degrees (science, engineering, mathematics) and who could grow to be MCCR personnel with critical skills are generally not in job assignments that use the requisite skills of the job classification. Too often the technologically demanding jobs are in the civilian technical support organizations. A pertinent question makes this point clear: how readily could a military technical officer go to work for an industrial contractor as a software or systems engineer? In general, we think that the answer is: not so readily. Then the question becomes: how will such personnel participate as technologically sophisticated managers in the acquisition process or as team colleagues with equal or better technological expertise in software development and PDSS projects? Job assignment sequences are needed that take into account the competition elsewhere and the level of technological expertise necessary to communicate effectively and equally in tri-sector (DoD military, DoD civilian, DoD contractor) organizational arrangements. Presently, the command/management director is far too often at a technological disadvantage because of the job assignment structure.

A distinct but equally important issue is that of career ceilings for MCCR personnel. While 23% of authorized personnel in the 49 series (492x -> 491x/499x)[19] are at the Lt. Colonel and Colonel rank; there are only 5% in the MCCR 26, 27, and 28 series at those ranks (2625, 2736, 2885).[20] These data complement the findings in the Beam Report [Beam 89] and provide a reason why software personnel, at least in MCCR jobs, cannot see a clear promotion path—there isn't one. Thus, for technically capable and astute MCCR personnel, the only reasonable behavior is to move out into another career path or to leave the service. Given the increasing importance of software personnel to industry in remaining competitive and the increasing importance of software in mission-critical projects, the promotion structure involving these critical skills needs serious reconsideration.

2.3. Labor Mobility

The approach used to examine mobility moves beyond the conventional focus on new graduates to include significant *inflows* into science/engineering computer-related jobs by people already working in the labor market. We also note that science and engineering loses a significant proportion of its work force to management and to jobs not involving science and engineering; these *outflows* must also be taken into account. Two levels of labor mobility are discussed: interoccupational (computing specialists, other science/engineering fields, management, other non-science/engineering jobs) and inter-sectoral (DoD industry, non-DoD industry, DoD civil service, non-military sector[21]). The interoccupational data pertain to both intraorganizational and interorganizational labor markets, while the inter-sectoral data pertain to interorganizational labor markets. The primary data are two NSF surveys of a

---

[19] The 491x and 499x authorizations include many communications-related positions (the old 30xx career field) and are not available or desirable to MCCR officers.

[20] The calculations are based on data from Chmiola. [Chmiola 89]

[21] The non-military sector is made up of non-DoD government and non-profit organizations.

national sample of experienced scientists and engineers in both the 1970s (1972-78) and the 1980s (1982-86).

2.3.1. Major Inflows to and Outflows from Computer-Related Jobs

To establish the baseline for the mobility of computer specialists,[22] it is important to understand the massive turnover in science and engineering in the United States. Between 1972 and 1978 over one-fourth (26%; 216,000) of all experienced scientists and engineers (831,000) had changed to a different major occupational group.[23] More important, 21% or 174,000 were no longer scientists and engineers. They had either become managers (14%, 117,000) or left science and engineering entirely (7%, 57,000).[24] These massive changes occurred in a six-year period, underscoring the significance of the dynamics of labor markets when talking about labor supply. It is not simply a matter of adding new entrants to the current distribution of workers. These distributions are changing, even without the new entrants and these changes involve significant proportions of the scientific and engineering work force.

In the labor market for computer specialists, embedded within the overall national labor market, there is even more change.[25] Over one-third of the experienced computer specialists moved out to a different major occupational group between 1972 and 1978. The bulk of these, 20%, went into management; an additional 6% left science and engineering entirely.

As for inflow data during this period, the data are incomplete, since they are prospective panel data from a 1972 sample of experienced scientists and engineers. Thus, the data miss the populations entering science and engineering from non-science and engineering jobs, including those who had left science and engineering previously and were returning.[26] Still, two large inflows from other science and engineering fields are identifiable. The greatest source of internal inflows are from mathematical specialties and engineering. There are inflows, though much smaller in magnitude, from each of the other science fields; the largest of this set is from physical science. What these data imply is that one must look beyond the degree and the field to identify the wider range of supply that is available as indicated by labor market behavior.

Since these data are for the 1970s, do they speak to the 1980s, or for that matter, to the 1990s? The 1982 NSF sample of experienced scientists and engineers has four years of

---

[22] These include computer programmers, computer scientists, computer systems analysts, computer engineers, and other computer specialists.

[23] The occupational groups are physical sciences, biological sciences, mathematicians, computer specialists, psychologists, social scientists, other scientists, engineers, managers, and other non-scientists/engineers.

[24] These calculations are based on Table C1 of NSF80-317, p. 16. [NSF 80]

[25] NSF80-317; Table C1. [NSF 80]

[26] We will provide some information on the return flows using the 1982 sample.

data available, to 1986, and the findings for computer specialties are quite robust for both the 1970s and 1980s. During this four-year period of the 1980s, 25% of all experienced computer specialists changed to a new major occupational group:

- 7% moved to engineering or another scientific field.
- 12% went into management.
- 5% left science and engineering altogether.

If we exclude computer programmers, the proportion staying in computer specialties is almost identical, dropping only from 75% to 74%. Since these results are not due to the inclusion of programmers, they hold as well for higher skilled software personnel.[27]

While there remains a bias for inflows from non-software and non-engineering jobs, the 1982 sample yields the following sources for the 1986 distribution of computer specialists: 70% had been computer specialists in 1982, but 30% had been working in other fields as follows:

- 14% had entered from engineering.
- 4% entered from other scientific fields.
- 6% percent entered from management.
- 6% percent entered from other non-scientific engineering jobs.

One important implication here is that some scientists and engineers who move into management or leave science and engineering will return. While conventional counts of new graduates indicate a substantial shortage in the supply of computer specialties, there is more reserve in the U.S. labor market than meets the eye and there is considerably more flexibility than is generally known in the experienced work force. Policies that are aimed at labor shortfalls or the gap between supply and demand need to be more informed about this labor market reserve and of the nature of its behavior.[28]

What is the impact of experienced science and engineering inflows and outflows on computer specialists? Are more leaving than are coming in? How is this changing over time? Comparing the 1970s and 1980s, it would appear that the situation is improving. The 1972 sample experienced a 22% net loss of computer specialists from the cohorts of experienced scientists and engineers (1972-78). In contrast, during the 1980s, there has been a net growth of 8% in computer specialists from the 1982 cohorts of experienced scientists and engineers. These findings indicate the importance of the overall experienced labor supply, including scientists and engineers outside of the software field *and* scientists and engineers who have left the field, but may return.

---

[27] Source: NSF public-use tape for 1982-86 national sample of experienced scientists and engineers in the U.S.

[28] For more detailed analyses dealing with specific critical skill areas and DoD industry, imaginative ways of utilizing corporate data and/or new data collection will be required to better inform the labor reserve interaction for software personnel, especially from the employer perspective.

Recent labor market analyses indicate occupational mobility is quite high for middle-aged and older workers who remain in the labor market, and much more akin to mobility for younger workers than heretofore thought. [Bluestone 89] In addition, within the U.S. Civil Service, the internal renewal rate[29] had no age differences within internal labor markets. Both of these findings highlight the flexibility and adaptiveness of the work force at all ages. When analyzing labor supply, much more attention may need to be given to workers already in the labor market, across a much wider spectrum of jobs and across a much wider age spectrum, than is conventionally considered.

2.3.2. Inflows to and Outflows from the DoD Industrial Contractors

An important question when studying capacity is the degree to which the set of DoD industrial contractors form a separate labor market that can be treated as independent of non-DoD industry in particular. Hence, we now focus on the movement of scientists and engineers between the following four sectors: DoD industrial contractors, non-DoD industry, the DoD portion of the civil service, and all other government and non-profit organizations. Emphasis will be on the flows of experienced scientists and engineers into and out of the set of DoD industrial contractors. Table 2-1 provides the movement rates from 1982 to 1986. The rows sum to 1.0, and reading across each row, the numbers provide the proportions staying (the diagonal element) or moving elsewhere between 1982 and 1986. For instance, row 2, column 2 indicates that 89% of the experienced scientists and engineers working in non-DoD industrial corporations in 1982 were still there in 1986. Column 1, however, shows that 6% of them moved to the DoD industrial sector by 1986 and another 5% (column 4) went to the non-DoD government or non-profit organizations.

Perhaps the most significant finding is that over this relatively short period of time, DoD industrial contractors lost one-third of their experienced scientific and engineering work force, with almost 30% moving to non-DoD industry. (See Table 2-1, row 1, column 2.) The size of this outflow is quite surprising, and certainly needs a much more in-depth examination. The sample size presently available, however, does not allow more detailed analysis by both scientific field and sector. Nor do we know the reasons for these large outflows. For instance, to what extent is it because of the volatility of the contracting business, which triggers moves from corporations who lost out on the contract bidding process?

In terms of the number of experienced scientists and engineers at work, for this sample the non-DoD industrial sector is almost nine times the size of the comparable DoD work force for industrial contractors. Thus, in Table 2-1, the second feature of significance is that in terms of relative numbers of exchanges between these two sectors, the two flows are roughly equal.[30] There are two major implications from these observations. First, the labor market of DoD industrial contractors is not nearly as segmented from non-DoD industry as is

---

[29] Out of the current population of jobs, the percent having new incumbents per year.

[30] DoD to non-DoD flow = 717 and non-DoD to DoD flow = 913; i.e., in this time period, a growth period for DoD industry, the flows into the DoD sector are somewhat higher than the outflows to the non-DoD sector.

Table 2-1: Movement Rates of Experienced Engineers and Scientists Between the DoD and non-DoD Public and Private Sector Jobs (1982-1986)

generally perceived. Hence, when questions about national capacity arise, it is indeed the national labor market—both DoD and non-DoD sectors—that must be examined.

The second implication is that even though the exchanges between the DoD and non-DoD industrial sectors are approximately even, because of the much smaller relative size of the DoD contracting sector, the proportional outflow from the set of DoD industrial contractors is massive. The following consequences could be quite important for capacity. First, the greatest shortages, at least in the software arena, were among experienced personnel in critical skill areas. The NSF samples represent experienced science and engineering personnel. If they also represent critical skill areas in software, then these outflows are extremely important. Second, the outflows may represent the shrinkage of firms X, Y, and Z while the inflows pertain to growth in firms A, B, and C; to the extent that experienced critical skill labor is moving out of the set of DoD contractors, this would be a contributing factor to the capacity problem. Alternatively, to the extent that experienced scientists and engineers from non-DoD industry can apply their skills and experience upon entry to the set of DoD contractors, it is less cause for concern. Given the different production environment; the different demands of much of MCCR products in terms of embedded, real-time systems; and the fact that Ada is not a dominant language in non-DoD industry, serious questions arise about the extent to which the experiences and skills of non-DoD scientists and engineers are immediately transferable. To the extent that there is a gap between those leaving the DoD sector and those entering it in terms of the operational and application domain expertise, for example, then some capacity is lost and additional time is required to regain it. While present data do not allow one to definitively form conclusions, the nature of the outflow raises an important question regarding labor and capacity. Given the size and complexity of future projects and the likelihood of a greater number of corporate players per project, it will be even more important to be informed about the transferability of skills between the two sectors and of the changing capacity of entire sets of DoD firms.[31]

---

[31] Also pertinent is the volatility of an individual firm's capacity over time.

3. Major Impacts of Other Factors on Capacity: A Systems View

As discussed in Chapter 1, increases in numbers of software personnel—even those in critical skill areas—will not be sufficient to meet the projected software demand. From both secondary data sources and our own primary data sources (executives surveyed and interviews), it was repeatedly stressed that the software capacity problem entailed more than "labor." This chapter addresses two other major factors. The first two sections cover organizational and technological impacts on capacity. The third section discusses a longitudinal systems approach for gauging national software capacity that has, for the most part, been missing from the studies and databases that are currently available.

3.1. Organizational Impacts on Capacity

Below we discuss three organizational impacts on capacity, all of which are interorganizational. The three pertain to requirements specifications and changes, contracting environment, and major aspects of program complexity. The latter is included here because it shifts the main problem from technological to organizational issues.

3.1.1. Requirements Specification and Changes

It would be hard to overstate the importance of requirements definition as a source of DoD problems in acquiring software. Boar [Boar 84] estimated that 60% to 80% of all systems problems are caused by inaccurate requirements definitions. Brooks [Brooks 87] argued persuasively that:

> The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements ...[n]o other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later....Therefore, the most important function that the software builder performs for the client is the iterative extraction and refinement of the product requirements. For the truth is, the client does not know what he wants (p. 17).

The Beam report [Beam 89] also reported the following: "Most problems that surfaced through software shortcomings were really due to immature requirements or impractical acquisition constraints" (p. 1).

In addition, during this study both industry and government executives identified requirements specification and their changes as the two most important factors contributing to the failure of military system development contracts to meet schedule or costs (see Table 1-6). Both interview data and other major secondary studies also stressed the significance of requirements specifications and changes.

While there is very high consensus on the importance of requirements specifications and changes, there is no such uniform agreement on the reasons cited and recommendations for change. Most of our sources indicate that the process of system acquisition—of which requirements specifications are but a part—is primarily responsible. Nevertheless, more specific causes are discussed below.

Too often, DoD acquisitions personnel are viewed as inadequately trained. Further, performance evaluation of acquisitions personnel can be misdirected; for instance, they may be rewarded for adherence to the letter of requirements specifications or for meeting schedule, even though the system does not fully meet operational readiness. In addition, given the military practice of frequent reassignments, larger projects spanning several years are susceptible to discontinuity in acquisition supervision. This in turn leads to loss of institutional memory, reducing the effective interaction between DoD personnel and contractors.

When the DoD has attempted to supplement its personnel with consulting organizations to support engineering and/or acquisition, one of the consequences is the inclusion of an organization (or a subcomponent of one) that has the contractual responsibility of generating comments on requirements specifications, contract proposals, requests for changes, and so on, with none of the responsibilities for maintaining schedules, budgets, or system performance. This is *not* to argue against such organizations; given the current personnel policies in the DoD, the need for some external technical assistance in the acquisition process is justified. It is to argue that such institutional arrangements are structurally bound to cause difficulties in system acquisition.

It has also been noted that end-user involvement in the specification process is at too low a level, is too late in the process, or is interrupted. In any of these cases, misspecifications that will affect the functionality of the system may result. Fixing these misspecifications, if caught in time, can cause schedule and budget slippage. Or, as sometimes happens, end-user problems may not be identified until after deployment, resulting in systems that are either unusable or under-used.

Additionally, in some cases there is an institutional tendency to overspecify the product at the outset. This tendency is especially problematic for unprecedented systems. A primary source of the difficulty is an inability to specify "what" is going to be built, without also detailing "how" it is going to be built. While there are some indications that the core of DoD STD2167 is reasonable in its attempt to specify a *process* for system specification, bid, and acquisition, there appear to be significant problems with the manner in which this process is implemented. The Beam report [Beam 89] pinpointed the weaknesses of the assumptions underlying DoD STD2167. Additional difficulties arise because of the degree of documentation and specificity required. This is especially debilitating when the project is large or unprecedented (i.e., risky).

While there are many genuine reasons for initial requirements specifications to change, if an escalating sequence of finger pointing for schedule slippage occurs, then the entire process may be derailed. At the other end of the spectrum, the process can also degenerate to the point that changes that are potentially beneficial to all parties are not being requested. A corporation may decide not to spend the time and effort necessary for certain waivers[32] because of anticipated reactions from the systems program office, or because of the number of comments the corporation will have to respond to from the contract support organization.

In summary, our investigations support the findings documented in other studies. Requirements specifications and changes in requirements represent a major problem area for the DoD and for the contractors. Virtually all elements—personnel, organizational and managerial, and technological—act to render this critical component of systems acquisition a major factor affecting capacity.

3.1.2. The Contracting Environment

In 1979, the GAO investigated software development contracting for ADP (automatic data processing) projects. Several "common causes of software development contracting problems" were identified including the following:

- Overestimates of the completeness of user requirements in the pre-contract stage.
- Inadequate criteria for contractor performance.
- Commitment to the total contract rather than the phasing of contracts.
- Inadequate management—too many changes, lack of inspection of intermediate stages of the work, and failure to require progress reports.
- Inadequate testing requirements.
- Failure to enforce contract clauses for recovery in the event of poor performance by the contractor. [GAO 79]

While not pertaining to MCCR projects, the managerial aspects of many of these problems have been addressed, with the exception of requirements specifications. In fact, our interviews indicated that perhaps enforcement has moved too far in the direction of micro-management.

In the intervening 10 years there have been a number of studies faulting the contracting procedures for software. Perhaps the most pertinent is the GAO analysis of the program to develop the Space Defense Operations Center (SPADOC). [GAO 89] They noted that the program cost estimates have grown from $290 million to $437 million, that completion is now planned for 1994 rather than 1988, and that the program is still far "from meeting its operational capability." The GAO claims:

> ...the Air Force...has accepted and paid for a system that is only marginally useful, does not meet most contractually specified performance requirements, and is not yet operational but which, according to the U.S. Space Command, when operational will offer some functional improvement over the current, primarily manual system.

---

[32] Even when such changes might result in some improvement in schedule, cost, or performance.

The GAO concluded:

> ...there are two primary causes of SPADOC's problems. First, the program is highly complex and technically risky...we know of no mission-critical controlled mode system similar to SPADOC that has become operational....Second, the Air Force did not prudently manage the SPADOC effort, given the technical difficulties and risks involved. System requirements were not adequately analyzed at the outset to identify which were most difficult to satisfy and posed the greatest risk to project success, and management strategies were not formulated and executed to accommodate these risks effectively.

As might be expected in the MCCR domain, the 1989 report acknowledges the difficulty of the technical problems and the contribution of technical difficulty to program management problems. Yet the GAO still maintains that with better requirements definitions and tighter bureaucratic controls, the Air Force could have avoided the problems. The Air Force responded that they spent four years on the systems definition, consciously staying within the "state of the art" and that, except where it would have required a major redesign of the system, they managed the contractor's efforts against key project milestones.

There is growing recognition, even with watchdog agencies such as the GAO guarding tax dollars, that software is somehow different from other things that the government buys and that it may call for a completely different philosophy and a completely different set of acquisition procedures. Brooks described the underlying fallacy in diagnosing problems and in devising solutions for software acquisition processes:

> Much of present-day software-acquisition procedure rests upon the assumption that one can specify a satisfactory system in advance, get bids for its construction, have it built, and install it. I think this assumption is fundamentally wrong, and that many software-acquisition problems spring from this fallacy. [Brooks 87]

This fallacy is widely held. It is, in essence, the common conception of good business practice; it implies a set of procedures that would seem to ensure good value for the dollar. It is simple, direct, and explainable to members of Congress and constituents. This conception suggests that the failures are caused by managerial laxity and points invariably to elaborated bureaucratic procedures and more fervent micro-management as the solution. However, this may be exactly the wrong approach for commodities like software.

A number of recent reports have done an excellent job of describing problems with the current contracting environment and devising proposed solutions. [ISSI 89] [Cheney 89] [BrooksReport 87] From our interviews and analysis of secondary sources, we agree with much of what they have to say and will not repeat these conclusions; a restatement of the inappropriateness of firm fixed-price contracting and of control procedures built on the "waterfall model" would contribute little. We will therefore touch briefly on two effects of the contracting environment that appear most important in lowering the productivity of software engineering personnel.

The emphasis in the acquisition process on traditional bureaucratic management tools like schedules and milestones leads to significant goal displacement. The schedules and milestones become the objectives of the program in place of the system and its attendant purposes and characteristics. Those closest to such an acquisition, however, know that there is much less reliability in estimating schedules and milestones until the program is very far along and requirements stabilize—until what is being built is finally known to a level of reasonable specificity. As Brooks noted "...the most important function that the software builder performs for the client is the iterative extraction and refinement of the product requirements." The current contracting environment, at best, undervalues that contribution and, at worst, denies it.

Goal displacement may also directly affect the military and contractor's ability to attract and retain the best software engineering talent. The best people want to spend almost all of their time designing and developing systems. They cannot do this, however, when designing and developing systems is no longer the primary goal of their organization; that is, when their organization is in the business of preparing winning proposals and of meeting schedules and milestones to the letter of contracts. The incentives lead to distortions. For example, a wizard at satisfying 2167A documentation requirements is at least as valuable to contracting organizations as a wizard in designing systems.

A related source of serious problems is the military role in managing the acquisition. It is extremely rare that the military personnel in System Program Offices (SPOs) have the necessary technical expertise and experience with the specifics of the application to really "work the problem" with contractors. The military role, the part that has not been delegated to civilian personnel or to technical support organizations, is most often reduced to managing the bureaucratic details of contract administration from an adversarial stance.

The level of the military technical expertise with respect to software engineering is seriously deficient relative to contractors and to the technical support organizations. There are too few technically expert military personnel. The military is not developing deep software engineering expertise in career officers and enlisted personnel; career management and assignment processes, wholly defensible on other criteria, preclude the requisite sequence of specialized technical assignments.

Defense contracting is a business of feast or famine. Funding fluctuations have numerous effects. One adverse effect is instability in software development teams. Turnover is very expensive on budgets and time schedules. It is not so much that it is difficult to recruit new team members, although recruiting is often a problem, but that even with recruit replacements, it requires substantial time and training to get the new staff "up to speed" on complex, long-lived projects. Formal training and experience on other projects help. But the new staff need to know idiosyncrasies of a specific project's substance and management before they can be very useful on it. Unfortunately, documentation is generally behind and often impenetrable. The weakest part of most software documentation, according to PDSS studies, [DoD 88] is on design decisions that rationalize the architecture. This documentation would, of course, be most useful in training new staff. Training on the current system usually falls to the existing software staff who are the only ones with sufficient knowledge. Managers face a dilemma in training. They can assign their best staff, thus losing some of their direct value on the systems effort. Or they can assign weaker staff, which reduces speed and quality in training, slowing the integration of new staff. Turnover is pernicious for programs of 8 to 12 years in duration.

Funding fluctuations are also one of several features of defense contracting for which companies charge a premium. While it is difficult to do a "comparative shopping analysis" because of the incomparabilities between DoD and civilian software projects, there is a common perception that DoD pays premium prices for its software. To some extent contractors, particularly small- and medium-sized contractors, must have this premium from times of feast to retain their key people in times of famine.

Another feature of the contracting environment exacerbates the software engineering labor problem. The contracting environment makes developing systems for DoD a much less attractive day-to-day occupation than it should be for the technical elite who are critical to developing successful systems. As Brooks [Brooks 87] noted, "Study after study shows that the very best designers produce structures that are faster, smaller, simpler, cleaner, and produced with less effort. The differences between the great and the average approach an order of magnitude."

Consider the following scenario. The most creative systems designers and architects find themselves spending a large portion of their time on activities not on the critical path to completing the system and not intrinsically interesting. They must prepare briefings for the next milestone, report to "software managers" whose primary technical expertise is hardware a generation or two old, and contribute to voluminous documentation. What the technical elite want to do is design systems to solve interesting problems. The DoD has the most complex, interesting problems. But the elite do not get to spend as much time directly on those problems as they would like because of the contracting environment. These technical elite are in short supply everywhere; they have options. They exercise options so that they can spend more time on interesting design tasks.

One serious alternative to micro-management of MCCR projects is for the DoD to shift the basis of competition for contracts from the substance of projects to the capabilities of prospective contractors. Where the basis of competition is on the substance of projects, the incentives are for a winning proposal, not for "accurately portraying the size of the system and the effort required to build it." [ISSI 89] DoD, Congress, and all of the oversight agencies must explicitly recognize that in acquiring software, the government is not buying a pre-specifiable product, but services that will lead to a product. This explicit recognition must translate into acquisition process details.

Use of specific methods to assess the capability of contractors [AFSCP 90] is another positive step in shifting acquisition emphasis during source selection away from the price and detailed nature of promised products to overall contractor capacity to develop software-intensive systems. It will be a more effective step if the review centers on the capabilities of the key people like software managers and on the actual track record of the organization in producing complex software. The one area where this type of alternative needs more consideration, however, is in the entrance of new firms to the process.[33]

Improving the contracting environment for software could substantially enhance the "national capacity" to produce software. Many of the gains will be realized through improvements in the ability of defense contractors to attract, develop, retain, and use personnel with the necessary critical skills.

3.1.3. Program Complexity

As indicated in Chapter 1, the recent surge in the size, scope, and unprecedented nature of ongoing and planned MCCR projects represents a qualitative change in the software development environment. The concept of program complexity, while difficult to pin down in quantitative terms, has a very clear-cut set of characteristics. First, program complexity appears to grow substantially with project size and the degree to which the project is unprecedented. Second, as the program grows in size, the number of people who can comprehend the program as a whole and hence hasten the proper development of the project is significantly reduced. The projects can simply be too large for any one or even a small coterie of individuals to develop the necessary depth of knowledge and familiarity required of an expert. Third, as projects become larger and more complex and are broken down into larger numbers of small modules for efficient solution generation, the system integration phase of the project becomes dominant and increasingly difficult. The number of possible configurations of the system becomes so large that effective integration and testing becomes the primary choke point. This problem is obviously aggravated as the number of prime to subcontracting relationships increase. Fourth, as projects grow in size and complexity, attention appears to shift increasingly from technical issues to organizational and managerial issues.

More specifically, as projects increase in size and complexity, the scale of internal teams (the number and size) begins to escalate to the point where team and project management becomes a significant problem unto itself. In a similar vein, the scale of prime to subcontracting relationships will change. Since subcontractors are bound by the same set of contracting rules as prime contractors, this increase in scale leads to proportionally greater management and coordination problems.

With the qualitative change noted above, experience becomes less reliable in a variety of phases in the project life cycle. Estimates of project size, time-to-completion, and cost become more unreliable. Solution strategies and software architecture become unusable. Experience of personnel, instead of being significantly useful, could become counterproductive.

The DoD could be faced with an ineluctable dilemma: its appetite for large and complex software projects appears to have reached a level where the problem is inherently impossible to solve within the specified technical, time, and cost constraints imposed on contractors. That is, given the state of technology, projects as currently defined may be impossible

---

[33] For discussion on this issue, see Chapter 3.2.

to do. If so, the technology, time, or cost constraints that are imposed need to be relaxed to bring the project into the domain of the achievable in the short to medium term.

Two possible approaches to ameliorating the capacity problem in the immediate future arose in our discussions with industry experts and are also mentioned in the relevant literature. The first, an organizational approach, is for the DoD to explicitly adopt an incremental approach to new, large, unprecedented systems. Provided that future enhancements are explicitly designed into the hardware, software, and systems integration "components" of the system, the argument is that such an incremental approach could radically alter the achievability of project goals in the immediate term.

The second approach is technological. It requires the DoD to recognize that, whatever the merits of Ada, there is much to be gained (for technology, management, and human resources) by considering software that does not follow DoD standards. Also, increased use of commercial off-the-shelf software might alleviate some new development. While there is some acknowledgement of this fact, a more open acceptance of such change through standards could be of immediate benefit to the entire MCCR acquisition and deployment aspects of the military.

In sum, the scale and scope of recent DoD MCCR projects are such that they qualitatively change the very concept of software capacity. Solutions based on personnel, organizational, managerial, or technical changes that can be reasonably anticipated are, at best, in the middle term. In the long term, however, solutions to the software capacity problem may require major technical breakthroughs in systems design, hardware, software, and systems integration.

3.2. Technological Impacts on Capacity

In this section we will briefly examine hardware and software resources and hardware/software integration.

3.2.1. Hardware Resources and Hardware/Software Integration

There are a variety of reasons[34] for insufficient access to hardware computing resources:

- The development cycle for projects with simultaneous development of hardware and software.
- The uncertainty of recouping investment costs over multiple projects.
- The specificity and relative paucity of the host and target hardware.
- The specialized nature of much of the hardware developed and/or used in DoD projects.
- The outdated status of the technology used in many DoD projects.

---

[34] Not all of these reasons are applicable all of the time.

Three human resource consequences follow immediately from the hardware resource environment of DoD projects. First, since the hardware is highly specialized and not in common use, or at times is old technology, it is difficult to identify a pool of computing professionals with the requisite experience and expertise in the non-DoD sector. Second, developing expertise on such hardware represents specific human capital that moves the cost of these expenditures from individuals to institutions. However, given the turnover rates in this labor market, profit-making institutions invest too little in such human capital, thus reducing the overall experience base available to the industry. Third, since the hardware may not be up to date, attracting and keeping young scientists in this sector is made more difficult.

A more subtle problem arises from the relatively recent shift from projects that were primarily hardware-based to projects increasingly dependent on software. Senior scientists (especially systems engineers) at most organizations come from hardware backgrounds, which may lead to two consequences: first, there may be an overt dismissal or downgrading of the technical difficulty and organizational status of software problems and hence software personnel; and second, there may be little or no communication between hardware and software engineers during the crucial concept definition and design phases of a project.[35] There are also claims that hardware-based engineers may not understand the subtleties of software technology, leading to impossible specifications (i.e., ones that cannot be met) or counterproductive levels of detail in the specification. Yet another claim is that such engineers save difficult problems for software engineers. If, over time, those with software backgrounds rise to senior status, this problem will be ameliorated. Right now, it is a difficult human resource and organizational problem.

Another consequence of the hardware specified is that contractors, and hence the DoD, cannot rely on the marketplace to generate technically sophisticated solutions to hardware and software problems. No individual contractor could sufficiently recoup the necessary investment. This requires that the DoD carry out both an explicit program of investment in generalized solutions and a technology transfer program to ensure the dissemination and adoption of technologies thus developed. While the DoD does have such programs, the size and scope of such efforts needs to be reevaluated in light of changes in the technical marketplace; in particular, the erosion of the once dominant position of the DoD as the employer of highly sophisticated technical personnel and development is an important change.

Where hardware and software development are undertaken in parallel, several technical problems compound development difficulties. First, hardware development could actually delay software development since software developers would have to wait for the relevant hardware. Worse, it could render software development useless until some key milestones (such as hardware availability) are met. Second, adequate unit tests may be difficult or impossible when hardware development is out of phase with software development. Third, system integration tests could represent prohibitive logistical complexity—i.e., simply organ-

---

[35] This has been described as hardware-based systems engineers "throwing the design specifications over the wall to the software engineers."

izing these tests with inadequate numbers of hardware units could render this phase very difficult. Recalling that system integration has been identified as one of the most difficult and key phases of a project in purely technical terms, this could be a significant factor in overall system quality, or lack thereof, and time and cost overruns. Fourth, if there is a paucity of development hardware, whatever the cause, *subcontracting* can be significantly hindered—thus reducing a major strategy organizations might use to cope with the general DoD contracting environment (such as project life cycle and funding uncertainty, etc.).

Another problem lies in the nexus of technical and organizational issues: the current practice of demanding specified levels of reserve capacity (such as n% of unused main memory or CPU capacity) could place unnecessarily tight bounds on the performance of software, thus artificially extending development risks, costs, and time. The argument that this reserve capacity is required for future enhancement misses the fundamental point that such requirements could negate the existence of future enhancements: projects could be delayed or could underperform to the point where the future enhancement phase is consumed by the development phase. Use of commercial processors, which tend to be smaller, lighter, etc., instead of specialized DoD hardware could ameliorate this problem.

By developing new or using highly specialized hardware, the DoD is unable to take advantage of economies of scale in computing power. There is some point at which the state of the art is reached, beyond which it will cost more, not less, proportionally to increase computing power. A worse possibility exists: when the state of the art is extended to meet high-performance requirements in a highly constrained environment.[36] Grosch's Law may well be reversed: increases in performance come at exponentially increasing costs.[37]

3.2.2. Software Resources

At least three distinct threats to capacity arise from the software resource base. First, the specialized nature of many DoD projects requiring specialized languages (at least in the academic and commercial worlds) leads to a lack of adequately trained or experienced software personnel, inadequate libraries, compilers, debuggers, design concepts, etc. Insofar as the use of Ada mitigates this circumstance, we should expect these effects to decrease. However, Ada itself comes with its attendant set of problems, arising primarily because of its relative immaturity, which we expect will ease over time. At present, it has been alleged that whereas use of other languages offers one or two ways to do something, Ada offers designers and programmers seven or eight options to investigate in selecting "good" solutions to design problems. Thus, effective use of Ada calls for experienced, expensive personnel.

The second problem arises from the lack of knowledge about existing software tools. Evaluation of these tools is costly, and the tools themselves might be costly to the point where

---

[36] A highly constrained environment includes: a narrow user base, strict limits on use of available capacity, security constraints on information dissemination, etc.

[37] Grosch's "Law" states that in a fixed period of time, a machine's "performance" is proportional to the square of the price. [Ralston 76]

their acquisition and use make sense only when the cost is spread over several projects, which is made difficult by the contracting environment. For similar reasons, developing such tools represents a very difficult problem for any single contractor. The current DoD thrust in funding the development of software productivity tools and active technology transfer could significantly increase the availability and use of productivity-enhancing tools and practices.

The third problem arises from difficulties facing software reuse. There is some evidence that considerable reuse currently occurs at the software unit level. What is lacking is the reuse of design concepts, architectures, and subsystems. As Kang and Levy indicate, technical, labor, and organizational/managerial hindrances to effective software reuse exist. [Kang 89a] [Kang 89b] Modules are not, generally, engineered for reuse in that they are not designed to solve generic problems; they are difficult to understand, expensive to modify, and poorly documented. The situation has not changed because developing reusable software is both technically more difficult and more costly. Given the contracting environment, few contractors could rationally arrive at the decision to invest in producing reusable software.

Enhancing reuse requires establishing standards. However, the development of such standards is directly dependent upon the degree to which the software has a large user base. Inherently, therefore, the DoD cannot rely on such standards evolving in any natural sense but must undertake an explicit program of standards development and implementation.[38] The development of an infrastructure to facilitate the flow of programs and related information between autonomous profit-seeking organizations is required. The infrastructure would need to support user access and the means for establishing, maintaining, and facilitating browsing through libraries to identify and retrieve reusable software. Two related issues to overcome are:

1. The legal liability if reused software is produced by another contractor.
2. Ensuring access to reusable software for small developers.

In short, the software reuse issue can be broken down into human resource difficulties, organizational and managerial difficulties, and technical difficulties.

Finally, it is also important to recognize that the current allocation of resources to technological aspects of software productivity could be disproportionate relative to the small proportion of the problem that is technologically based. We have considerable, though anecdotal, evidence that improvements in tools, debuggers, libraries, etc., while important, could have an impact upon as little as 20% of the software development process. The remaining 80% represents either areas lacking technological solutions (i.e., organizational, managerial, or personnel areas) or where technological solutions are inherently more difficult to achieve (such as the system integration phase or system design phase).

---

[38] Note that this observation applies as well to the hardware environment for DoD projects.

3.3. A Systems Approach: Changes over Time and Interactions Among Factors Affecting Capacity

Even a brief review of previous studies on software development generates an exceptionally long list of important factors that influence the national capacity to meet the software demands of existing and projected systems. In grappling with these factors that influence the nation's capacity to produce software, two separate issues need to be kept in mind. First, each factor is aggregated to an unknown, but directionally specifiable (i.e., worsened), degree by a project's size and complexity. That is, any given factor is made worse as a project's complexity and size increase.

The second issue pertains to the domain of possible solutions to the problems identified. In broad terms, the solutions that are applicable or have been attempted for each of these factors can be categorized into those involving labor or personnel issues, those involving organizational or managerial issues, and those involving purely technological issues. However, it is extremely important to keep in mind that the boundaries of these problems are permeable; i.e., the dividing line between any two of these sub-domains is not clear-cut and hard, and their study is not within conventional academic disciplines with well established paradigms. Nevertheless, this three-way division of the solution domain serves the purpose of clarifying the type, relevance, and range of applicability of suggested solutions to the software capacity problem and helps in identifying potential gaps in the current range of DoD funded research for ameliorating the software capacity problem.

Table 3-1 groups the three major factors in terms of time: those dealing with resources at any point in time and those characterizing the process moving over time. Issues can be characterized by the extent to which they are primarily labor issues, organization or management issues, or technological issues. They can also be categorized by the extent to which the focus of attention is primarily static or dynamic. Static analyses tend to deal with stocks or resource levels (size of labor force, average salaries, number of major contractors, availability of software tools, etc.). The entries in the individual cells capture our understanding of the current level of investment in assessing or solving problems of capacity; the smaller x denotes less investment than the larger X, and a blank denotes effectively zero investments.

The crucial boundary areas linking labor, organization, and technology are even less studied and we must rely almost entirely on expert opinion to obtain information on these areas. To understand these critical boundary areas, it is necessary to move beyond conventional static studies where one observes the system (or subsystem) at a single point in time and attempts to extract the interrelations based on the correlations at that time. That strategy will not work for the simple reason that these interactions are embedded in time and cannot be identified, let alone studied, by static investigations.

Just as MCCR projects are characterized by embedded, real-time systems, the capacity problem is embedded in time and is systemic. Consequently, the use of static data and analysis to infer what went wrong or the reason for the project being "broken" will almost

Table 3-1: Extent of Analysis Conducted on the Three Major Factors Affecting Capacity

certainly yield biased conclusions. It is essential to have information on the full population of projects or a representative sample from the entire range of projects. In short, the scanned environment needs to be more comprehensive (more variation) *and* equally important, viewed over time. How can one assess change, positive or negative, without time series data? Most importantly, the scope for this more comprehensive, longitudinal view needs to include all three of the major factors affecting capacity in order to view the overall system as it changes over time. Just as hardware/software integration is a strategically important feature of MCCR projects, the system integration of these three major factors affecting capacity—labor, organization, and technology—is strategically important for understanding national software capacity.

Dynamic analyses tend to deal with flows or processes (changes in labor force composition, births and deaths of minority owned contracting firms, rates of technological change, etc.). The contrast between these two kinds of analyses is shown by a few examples of each in Table 3-2.

Most of the work to date has dealt with static analyses except in the area of technology. Because technology appears to change so rapidly it is not too surprising that it has attracted a dynamic focus. On the other hand, because longitudinal data are particularly difficult to come by and because data definitions are often not stable over long periods, most studies have been driven by static or short-term analyses.[39]

Once a systems perspective is adopted, it is also important to identify the germane populations. For example, what are the current populations of projects in the specific MCCR appli-

---

[39] Note that a comparison of several years data does not, of itself, qualify as dynamic analysis in the sense conveyed here. Dynamic analyses are explicitly concerned with flow rates and the effects of those rates on stocks. Of course, multi-year comparisons are useful beginnings to dynamic analysis and often provide important insights into problem areas.

Table 3-2: Examples of Static and Dynamic Analyses

cation domains? In the ADP world? It is important to know these populations to calibrate what the specific smaller data set being examined represents. When dealing with DoD contracting firms, a systems perspective suggests that both subcontractors and prime contractors are important, as well as their past record of performance and expertise by application domain. In labor, as pointed out in Section 2.3, the scientific and engineering population is a national one, and sampling DoD firms provides only partial information. National samples, including non-DoD labor, are important to more fully determine labor potentially available and labor supply leaving the DoD sector. Other pertinent populations include the case tools, the hardware, and the set of firms in the U.S. It was indicated in Section 2.3 that there may be a substantial labor reserve. There may also be a much more substantial pool of organizations for producing MCCR software than is currently being utilized. What are the populations of subcontractors and how have they changed over time? These types of questions, derived from a systems view, permit a much more penetrating view of national capacity. It is indeed a national problem and it needs a broader national systems perspective to match it.

Below, we briefly discuss each of the three major factors affecting capacity within this systems view. Since labor has been a primary focus in prior chapters, here we give more attention to organizations and technology. For each of the major factors, however, a key feature is the interaction among factors.

3.3.1. Labor Issues

The investigation of issues related to software labor needs to be embedded within the overall context of the software industry as depicted in Table 3-2. While each column represents a separate domain, it is critical to recognize that they do not represent independent domains; i.e., the labor, organizational and managerial, and technical issues interact. In this section, we focus on labor issues with particular reference to the interface between these and technical and organizational/managerial issues.

Conventional human resource studies (wage and salary studies and policies, career progression, merit systems, recruitment and training, etc.), while germane, are essentially static. That is, they focus either on a single time point or look at a "panel" of people. By doing so, they miss the structural character of the industry, which results from the dynamic interaction of people, organizations, and technology over time. One can go further and argue that this structure *is* the dynamic interaction of the three elements. We term this structural character the labor market.

This labor market can be viewed at varying degrees of magnification: the internal labor market at the organizational level; the sectoral labor market at the government, commercial, and DoD levels; the regional labor market;[40] and the national labor market. Investigating the dynamics of these markets will enable the estimation of real size, composition, sources, and sinks of appropriate labor pools. For instance, while there is some evidence that the national labor market responds to changes in demand, how a market characterized by stiff technical educational requirements adjusts and over what period of time is not clear. As a consequence, potentially significant sources of skilled personnel may be left untapped or, at best, under-used.

3.3.2. Organizational Issues

> The lack of direct software experience (particularly in real-time systems) in key management positions results in inadequate methods and tools being selected for design....This same lack of expertise also results in unwise assignment of personnel to software modules where the level of difficulty is radically underestimated relative to the skill level needed for successful implementation. [ISSI 89]
>
> The current macro-organizational environment is not consistent with the functionality required by improved and appropriate system life-cycle processes. [ISSI 89]

Neither the management of DoD contractors nor acquisition organizations have escaped criticism as major contributors to software development difficulties. For instance, acquisitions managers have been characterized as overly bureaucratic with tendencies toward excessive micro-management on the one hand, and as too careless and prone to rely on contractor expertise on the other. The Defense Science Board report [BrooksReport 87] identified management as the biggest problem area for software development. The MCC Field Study [Curtis 88] included communication and coordination difficulties among key problem areas. The SEI Contractor Capability Assessment Study concluded that 98% of contractors are not operating under a qualitatively well-defined process for software development. Zelkowitz et. al. (1984) concluded that corporate management lacked understanding of software issues.

Our own investigations provide some support for these conclusions but we are reluctant to name management as the key problem area. Many organizational problems that influence

---

[40] Although the software engineering industry has characteristics important to a national labor market (it is highly professional and technical), it also appears to be somewhat geographically balkanized; i.e., it may operate as a mixture of a regional and national labor market.

capacity are not simply managerial. They are embedded in a network of influences that is often national in scope. These include:

- The contracting environment.
- Elements of the acquisition process.
- The state and availability of hardware and software.
- The enormous scale and complexity of the systems.
- The quality and availability of labor.

We cannot agree with the ISSI-Lockheed report that "...there seems to be a naive, implicit assumption on the part of the contractor and the government that satisfying the deliverables according to standard is a guarantee of success." (p. 36.) Our interviews with industry contractors and DoD acquisitions officers convince us that there is little naivete on either side. There is, however, frustration. There is occasionally a sense of being overwhelmed by the magnitude of projects and attendant reporting requirements. While there is finger pointing, there is also a strong sense that progress can be made in overcoming obstacles if both sides continue to hammer away at the problems. If there is naivete, it is in the belief that problems can be eliminated once and for all through improved guidelines and procedures. Private industry does not work from this premise. There is no reason to assume that MCCR systems development should.

An important conclusion of the Defense Science Board report [BrooksReport 87] is that DoD will continue to experience shortages in skilled personnel and should plan to live with them as best it can. We found evidence of the shortage but find it premature to conclude that the problem is permanent. Historically, labor supply has shown a remarkable ability to adjust to demand. However, it is the time period of such adjustments that is most uncertain. There are apparently large, untapped reservoirs of talent embodied in female and minority students and other employed people in the country. It appears that the structural barriers to gaining access to that talent may be less formidable than is generally believed. If the civil service can transform secretaries into software engineers and team leaders who develop real-time embedded systems, it is hard to understand why a concerted effort among government, industry, and academia could not significantly alter the number and quality of scientifically competent systems designers, software engineers, and programmers.

3.3.2.1. Populations of Organizations

Software for defense systems is created and maintained by hundreds if not thousands of separate organizations. Through reorganization, divestiture, acquisition, financial failure, and strategic decision, many firms come into and go out of existence every year. At present, we do not know the extent to which the nation’s capacity to develop MCCR (and ADP) systems is influenced by the changes in the populations of these organizations. We suspect that it is not trivial. Nor have we identified with any certainty the primary economic and ecological causes for changes in these populations. A longitudinal database linking organizational size, mission, date of birth, and—where appropriate—date of death to DoD contract activity, changes in the scientific and engineering labor force, and various macroeconomic variables is required to understand population-level influences on software capacity.

We believe that this understanding of population-level influences requires answers to the following types of questions. What kinds of organizations spring up rapidly to meet new DoD demands for systems? What are their characteristics? What are their life expectancies? To what extent do major DoD contractors spin off new organizations in the manner of the micro-electronics industry? What is the relative contribution of these spin-offs to total capacity? Over what period of time? How do changes in the distribution of the military service units among commands affect acquisition and PDSS of MCCR systems? How does the rate of change in the size of organization populations affect the supply of labor required for developing and maintaining MCCR systems? How do changes in the distribution by size and geographic location of contracting organizations influence the effectiveness of prime contractors and their major subcontractors?

If, as we suspect, the effectiveness of large prime contractors in delivering MCCR systems is largely dependent on the availability of high-quality subcontractors, and if the population of those subcontractors is rather volatile, then mapping and explaining that volatility is important to understanding the factors that affect the nation's software capacity. Similarly, to the extent that expertise in systems development resides not just in individual scientists and engineers, but in whole organizations, then the frequency with which those organizations come into and go out of existence influences capacity through its impact on the effectiveness of skilled labor, the confidence that other organizations have in their working partners, and the rate of technology transfer.

3.3.3. Technological Issues

This study's focus on technological issues is not on technology *per se*; i.e., the focus is not on issues such as the appropriate software engineering practices, productive software tools, and so on. Rather, the focus is on the study of the *effects* of technological change (and the rapidity of that change) on human resources and organizational issues and *vice versa*. In addition, special attention will be paid to the dynamic interactions among technology, labor, and organization/management.

The interaction between technological issues and labor leads to a variety of problems, ranging from the most useful educational training for raw recruits to the impact of DoD-mandated hardware and software standards on labor turnover and availability. Considering the highly specialized application domains of DoD software requirements, it is not clear that changing university curricula, in and of itself, would be a potentially useful source of meeting current skill shortages.

To obtain a larger supply in the critically short skill areas, the most appropriate sequence of job assignments, training courses, and project assignments for developing required skills in a shorter time frame needs to be identified. For example, would a DoD-sponsored, extended internship on a real development project for undergraduate science and engineering students reduce the time required to "grow" a skilled software engineer? What are the resource and productivity implications of such a program?

More generally, current shortages of skilled labor require that a variety of alternative technological strategies be explored to examine their impact on the shortage. For example, if software reuse were to be adopted on a much wider scale, does this change the types of skills required? Would shortages be worse in the short run, but improve as adoption increases? With the increasing adoption of Ada for DoD projects, coupled with slower adoption in the commercial sector, there will be an imbalance between the demand for Ada-skilled labor and the rate at which either academic or commercial institutions can foster Ada expertise. Hence, the institutional arrangements required to ensure the development of adequately skilled Ada programmers takes on significance. On a broader scale, the impacts of the DoD adopting *de facto* commercial standards on the supply of both raw and skilled labor needs to be evaluated in light of the purely technological disadvantages or advantages of the strategy.

In a similar manner, organizational impacts of technological changes require study to elucidate the range of effects brought about by the adoption of a given technology. Generally, the issue of the appropriate institutional arrangements to foster the investment of profit-making corporations in policies, procedures, and training not immediately conducive to profit-maximization requires a thorough understanding of the dynamics of contracting organizations in this volatile market. Specifically, what are the incentives and organizational infrastructures required to encourage the development and dissemination of reusable software? A related but different issue concerns the anticipated surge in software requirements for PDSS of deployed systems. What institutional changes are required to incorporate the post-deployment support of advanced MCCR systems as a major design criterion at the development stage? What is the potential impact of cross-development tools (these are tools that would allow software engineers and programmers to use the latest, most technologically advanced host for development even when the target hardware is quite different) on software productivity? Who would have to incur the cost of developing such tools?

Another area for investigation is the effect of "office-automation" technology as applied to the management of large-scale development projects. Given that project and team management are fast becoming, or already are (cf. Chapter 1), significant problems on DoD projects, the impact of such a technology on the structure of project teams, on the changes in communication patterns among and within teams, and on documentation needs to be understood and documented.

In summary, technological changes have, potentially, far-reaching consequences for labor and organizations. Equally, organizational and labor constraints could significantly retard the development and/or adoption of productive technological innovations. Thus, attacking purely technical problems with little attention to their non-technical effects could be ineffective, or worse, counterproductive.

Having now examined the major factors affecting capacity and the data that were available or could be collected during the short period allotted to the near-term study, in the next chapter we take a more in-depth look at the quality of currently available data.

4. Data Quality

4.1. Data Collected for the Near-Term Study

At the study's inception, a decision was made to limit the overall amount of new data collection because of the brief time frame and funding constraints of the effort. Four exceptions were made:

1. An executive questionnaire was developed and administered to 106 senior industry and government representatives to determine their perceptions of factors influencing DoD software capacity.
2. Senior managers, software and systems engineers, and human resources representatives in seven corporations were interviewed about recruiting practices, specific shortfalls of new or experienced software professionals, and career paths and retention of software professionals. They were also interviewed to gain a firsthand sense of the systems integration/hardware/software interface issues affecting software capacity.
3. Employment agency representatives (head-hunters) in the northeast and the northwest were interviewed to collect supplemental data about shortfall areas and salary differentials by geographic region.
4. Military and civilian government officials were interviewed about recruiting, assignment, retention, career paths, shortfall estimates, and recent studies of potential use to the study team.

We have referred to these data as "primary." It is information whose quality, reliability, and limitations we know firsthand. Questionnaires used for these primary data collection efforts are in Appendix B.

Perceptions that software capacity involves factors other than labor supply and demand were reinforced by the interview data. Several policy issues were raised, e.g., the role of current clearance policies on schedule slippage for classified projects and the impact of project size and complexity on government management and contractor/subcontractor relationships for projects in development now. Another important outcome of the primary data collection was to illuminate the ambiguous and inconsistent nature of most data available within organizations for use in estimating software development capacity.

Many kinds of information were treated as "secondary" by the study team. We classified as secondary National Science Foundation data tapes we acquired for use in analyzing the career moves of experienced computer scientists and engineers in the U.S. Other secondary information we used include: briefing text from contacts in the Air Force, Army, Navy, and various DoD offices; data from government studies underway; General Accounting Office and Inspector General reports; published articles in journals; corporate, proprietary documents about software capacity; and MITRE and other contract support metric data for specific projects. Secondary data sources used in this effort are too numerous to detail here and are presented in Appendix A. Some of the metric data and other contract support documents are identified only by application domain to protect the confidentiality of individuals and organizations, as promised, in return for information sharing. For the same reason, none of the corporate proprietary documents are listed. Data from classified projects were extracted and sanitized by MITRE.

As noted earlier, the focus of the near-term study is on MCCR applications. Secondary source data were used, however, to estimate the demand for software for both MCCR and ADP military applications and commercial demand as well, especially for Ada projects or products. Within the MCCR arena, data to estimate demand for new development versus post-deployment support were based heavily on DoD documents. The amount of information available is substantial, but the quality, comparability, and density of reliable information leaves much to be desired and will be addressed later in this section.

Empirical studies of a single factor influencing software capacity exist. [Krasner 87] Understanding the role individual programmer behavior plays in producing software systems does not, however, provide information about the relative contribution of various other factors, such as team size, prior working relationship of team members, relationships between prime-contractors and subcontractors, or ability to manage system integration tasks. Similarly, there are excellent cross-sectional studies looking at multiple projects or organizations, applications, and factors at just one point in time. [Curtis 88] Such studies provide a partial conceptual framework and a glance at software capacity, but lack a time dimension or systems perspective that would enable us to gauge changes in status of the software and total systems development for a large scale, multi-year MCCR project.

4.2. Constraints and Gaps in Currently Available Data and Database

Three serious conditions constrain our ability to do meaningful analysis and forecasting of software capacity. First, operational definitions and concepts related to software capacity are still being developed. Determining the number of "software engineers," estimating the size of software per system in terms of "lines of code (LOC)," or calculating the amount of "reuse" feasible becomes an untenable task when there are several definitions in use and no mechanism for reaching concensus or closure on operationalization. [Hefley 88] [Firesmith 88] Also, definitions change over time. For example, here is the metric for "experienced software personnel" used by one acquisition support organization in 1985: "Experienced personnel are nominally defined as those individuals with a minimum of 5 years experience in MCCR software development and a minimum of 3 years experience in software development for applications similar to the system under development." [Coles 85] In 1988, the definition shifted to: "Experienced personnel are defined as those individuals with a minimum of three years experience in software development for applications similar to the system under development." [Schultz 88]

The second condition is the insufficient quantity of MCCR project or commercial product information available. <u>There are no comprehensive compendia of either MCCR or commer-</u>

cial software development data, e.g., lists of applications, their development phase, estimated/actual size, estimated/actual personnel or costs. Also, it appears that neither the military[41] nor commercial organizations ever established systemic information collection efforts about either developing software-intensive systems or the post-deployment support of these systems over time. To produce an estimate about Ada demand presently requires consideration of several data sources, because any particular list omits multiple, large projects/product development efforts. We combined Ada Joint Program Office Clearinghouse data with briefing documents provided to the SEI by various branches of the military and published articles from *Aviation Weekly* to get a fairly comprehensive list of Ada efforts. Omission of any one of the documents would have resulted in variances of 10 to 14 million lines of code in the military systems estimate. A particularly important data element missing from most of the available sources is time. Two critical aspects of software capacity related to time are:

1. The relationship of other data, e.g., personnel requirements, to the software development life-cycle stage.
2. The relationship of the software component(s) development to the overall system development timeline.

The present lack of information relating timelines to line of code estimates or system development schedules is a critical data gap.

The third related constraint is the insufficient quality of data about software capability. Data accuracy is a serious problem. Collection of accurate information requires time and competence from technical professionals. For companies working under contract to the federal government, the cost of data collection is a part of their overhead. Government officials and acquisition support contractors interviewed for this study and other recent reports [Beam 89] indicate there is an increased tendency for the government to micro-manage software-intensive development projects. Asking for data about many aspects of a project on a frequent basis probably places more of a demand on contractors' overhead than they are willing to spend. Given the inability of contractors to refuse government requests for information outright, it is easy to envision the routine they may invoke: contractors provide the minimum information they believe they can get by with and they make little attempt to provide accurate information. Closely connected is the trend toward increasingly adversarial relationships between contractors and government customers. None of the parties wants to be held responsible for the large cost overruns, schedule slippage, or other problems plaguing some MCCR efforts. Thus, there are disincentives for both military and industry managers to report actual status of software development efforts when problems begin to occur. Also, at times there are incentives for contractors to exaggerate actual accomplishments to ensure payment for work underway. If a contractor reports schedule slippage or inability to meet requirements for a product, the government could withhold progress payments. Protracted situations of this sort are associated with loss of promotion or loss of employment among project members. [PropDat3 89]

---

[41] The Data Analysis Center for Software, RADC, does operate a database service about Air Force projects on a fee-for-service basis, but this database is not comprehensive.

Where data are collected about multi-year projects, turnover in acquisition support personnel and lack of consistent data collection over time limit the estimation accuracy within projects. Acquisition support professionals report that comparable data across projects are often not kept. Each System Program Office (SPO) or Program Management Office (PMO) emphasizes certain data of interest or expresses no interest in data elements collected as long as assurance of reasonable progress is reported. When performance problems arise, data collection and data quality become the focus of attention. A result of this phenomenon is that the best data are available for the projects experiencing acute difficulties, often involving litigation or audits. Information is unavailable for characterizing the population of projects in an application domain, a class of weapons systems, a branch of the military, etc.; or for enabling statistical sampling or providing a comprehensive and more accurate estimate of software capacity.

Many of the data gaps and constraints described above are not unique to the MCCR community. There are major constraints and gaps associated with the status of database development that affect software capability estimation for both the commercial and military sectors. For the domestic commercial world, there is no national or even regional or local database enabling producers to estimate the size or nature of their competition for a particular product. For the DoD, it seems almost prohibitively difficult to identify organizations capable of producing real-time embedded software analogous to MCCR systems where capacity shortfalls might occur. Development of such databases could serve to open up competition in the MCCR community and provide new markets for the U.S. software industry.

For the military sector, information queries within a division of a command, across commands within a service, or for tri-service information about comparable application efforts cannot be handled at this time. Simple management inquiries within a division about cost and schedule slippage on major projects require access to multiple databases, and the lack of comparability of data elements makes accurate aggregation difficult. For example, it would seem potentially useful to be able to access and analyze the personnel metric data from AFR 800-43 for indications of critical shortfalls during the next decade. While the Joint Integrated Avionics Working Group (JIAWG) and each service involved in large-scale avionics and electronic warfare, Ada software, and systems development projects are responsible for delivering these crucial unprecedented systems, the data elements and database to support management are not in place. We believe along with others [Beam 89] that the operational readiness of the U.S. military forces is adversely affected by the current state of information management.

Chapters 5 and 6 of this report describe long-term study contributions and initiatives the government can support to address the current data quality constraints and gaps.

5. Long-Term Study Contributions

As expected from the beginning of the near-term study, but now driven home by initial assessment in the prior four chapters, it is clear that the United States has a software capacity problem. Moreover, this capacity problem requires attention at the national level:

- It is a national problem since it bears directly on our ability to produce and maintain mission-critical weapon systems for our national defense.
- The problem will not go away soon and will probably get worse.
- The nation's software capacity involves cooperation among the DoD, the U.S. Civil Service, and DoD industry *and* competition between these organizations in the DoD sector and the remaining private sector industry in the United States.
- Closing the gap between expected software demand and capacity to meet that demand by both labor and productivity growth will require more realistic timetables.
- To narrow the gap will require at least a three-pronged attack involving labor, organization/management, and technology. Over the next decade, at a minimum, serious gains must be made in all three areas.

It is not hard to conclude that the nation is facing a substantial capacity problem in being able to deliver mission-critical software over the next decade. Presently, there is a capacity problem and there is strong reason to expect that the situation may get even worse in the 1990s. To more accurately calibrate *changes* in the trajectory of that capacity over time will be a much harder task. Yet, it is the joint baseline and relative changes in that baseline over time that will be needed for macro planning and for inputs to decisions regarding trade-offs. The foundation has now been established with an overall framework and the data identified that are needed to begin this task. Given the importance of assessing the trajectory of the nation's industrial software capacity and of gauging changes in the set of primary factors which could affect that capacity over time, the approach will utilize two levels of attack:

1. Development of a macro-level national capacity indicator model.
2. Development of more "micro" modeling and analysis on major factors affecting capacity as inputs to the macro model so as to better refine it and calibrate anticipated directions of change.

It should be clear from Chapter 4 that substantial effort will be required in data collection and integration. The scope is intentionally national, involving DoD and non-DoD sectors, and the task requires reasonably consistent time-series data. In some areas of the DoD, partial baseline data are available and may need to be supplemented. Certainly, data from both the DoD and DoD industrial contractors will be required. Based on the near-term study, cooperation from DoD corporations has been extremely helpful and we expect this to continue to be the case. Synthesis and integration of tri-service data will also be necessary, as well as data across commands within a service. Additionally, collection of new data, especially representative samples of large, mid-size, and small DoD and non-DoD corporations, will be undertaken to provide information on relative demand, outside of the DoD in particular. The databases should therefore provide national coverage of four sectors: DoD (military and civilian), DoD industrial contractors, non-DoD industry, and non-DoD government.[42]

The macro-level national capacity indicator model will provide national estimates of the demand for software and of schedule slippage (at least for the DoD sector). These estimates will then be used in dynamic analysis and forecasts.

The dynamic micro-level analysis will cover three major factors affecting capacity:

1. Labor (organizational labor markets, national technical labor markets).
2. Organizations (impacts of the contracting environment and budget processes, organizational/managerial factors influencing software productivity).
3. Technology (technical factors influencing software productivity).

Each of these factors will involve four aspects:

1. Collection and synthesis of longitudinal data.
2. Design and exploratory analysis of the dynamics and interactions among the three factors: labor, organizations, and technology.[43]
3. Dynamic analysis and preliminary model development.
4. Input into the macro model.

---

[42] With special attention to the federal government and especially NASA and the FAA. For DoD, both MCCR and ADP development and maintenance will be included.

[43] Attention will be given to application domains, including those most analogous to DoD real-time embedded systems found in commercial industry.

6. Recommendations

Based on our preliminary findings, we conclude that the nation's software capacity problem is acute. Many of the conditions contributing to the problem are not new, and the magnitude of the problem appears to be increasing rapidly. To gain control of and improve the situation, Air Force leaders must be committed to *bona fide* changes in the way business is done among government, industry, and education establishments. The U.S. Air Force has an opportunity to take a leadership role and initiate national interventions to improve the situation.

Recommendations for action are divided into two parts:

1. Specific steps for improving Air Force software capacity within the service and within industry.
2. Recommendations involving broad federal government/industry interventions where Air Force leadership may be the key to moving beyond yet another study and on to real change efforts.

All of the recommendations will require government and industry leaders to make serious commitments to change.

6.1. Air Force Actions

Air Force leaders may consider taking action to implement some of the initiatives recommended below for their branch of the government. Specific recommendations are enumerated here about the organic capacity of the Air Force to manage software-intensive acquisitions and ways to improve estimation and monitoring of software capacity.

Estimating and Monitoring Software Capacity

Problem: The quality and availability of a set of even gross indicators of software capacity for the Air Force or the nation elude us right now. Estimating and monitoring software capacity is very difficult because of differences in definitions or metrics in use for essential capacity factors such as "source lines of code" and "experienced software engineers."

Initiatives: Two initiatives are recommended:

1. Support the development and use of a set of key capacity indicators in conjunction with organizations such as the SEI, IEEE, appropriate industry, contract support organizations, and government representatives.
2. Convene a working group involving business and senior technical representatives from government and industry to determine realistic costs and means for collection of data on the minimal set of key capacity indicators. A prior commitment would be needed to provide funds to compensate DoD industry contractors for data collection costs.

Problem: The quality of data about software capacity seriously limits our ability to estimate current performance for individual Air Force projects over time or to do any cross-project or program estimation of software capacity.

Initiatives:

1. Convene an Air Force-sponsored national meeting to create awareness about the software capacity crisis and the role inaccurate information plays in leaving the nation's government and industries at risk of making badly informed decisions.
2. Create a long-term strategy to gain commitment from senior leaders from each command, managers of senior contract support organizations, and industry executives to participate in efforts to improve the nation's ability to forecast software capacity.
3. Explore the feasibility of promulgating the use of a set of management indicators of the kind being developed in the updated *Air Force Systems Command Pamphlet [AFSCP] 800-43* for all new software-intensive MCCR projects throughout the Air Force. [AFSCP 86]
4. Conduct outreach activities to determine ways to improve data collection about analogous and relevant commercial industry software capacity information.
5. Design a small pilot effort to collect, from DoD contracts that are currently funded, the key set of software capacity indicators at various stages of system development, software life cycle, and for at least two application domains. Key features of the pilot would be:
    - Agreement by contractors to participate in training and provision of quality data to the SEI or another mutually acceptable neutral third party for use in national capacity estimation.
    - Commitment from the Air Force to compensate the contractors for costs incurred in the effort.
    - A critical review of the entire set of information currently provided by each contractor to the government with a goal of reducing the quantity and improving the quality and distribution of the information.
6. Take the set of key software capacity indicators developed under the previous initiative and install it in new Air Force contract-monitoring policies and practices.

Air Force Software Acquisition Capacity

Problem: Organic resources to manage software-intensive acquisitions are very limited by current assignment and promotion practices for both career officers and enlisted personnel with software experience or expertise. The difficulties of accurately identifying these people and of offering them a career path beyond captaincy lead to a serious problem in retaining them.

Initiatives:

1. Initiate a formal review of the impact of the 49XX reclassification on Air Force personnel.
2. Develop and publicize career paths or patterns up to at least the rank of Colonel for Air Force personnel, especially 26XX, 27XX, and 28XX series, performing in computer-related assignments.
3. Develop assignment procedures and practices to enable technical personnel with high performance to experience the maximum number of technical assignments and to be promoted into key acquisition management assignments.
4. Provide appropriate resources (time, funds, expertise, etc.), and especially senior Air Force sponsorship, for ongoing survey efforts to identify, track, and evaluate the effects of policy changes on promotion and retention of Air Force personnel with software experience.

6.2. Broad National Policy Considerations

Educational Initiatives

Problem: There is a serious shortage in the supply of U.S. citizens with systems or software engineering education and application-domain experience.

Initiatives: Two efforts are needed now:

- Organize knowledgeable parties, e.g., IEEE, ACM, AFCEA, AIA, to develop a program for industry use which would identify engineers and others for technological updating, and would support them through sabbaticals instead of early retirement or employment termination for technologically obsolescent engineers.
- Develop a tri-service career planning and scholarship program with explicit career paths in both government and industry for enlisted personnel and junior officers with application experience so they can enter graduate or continuing education programs in systems or software engineering and return to work in the MCCR community.

Problem: The supply of new graduates at the bachelor’s and master’s level in systems engineering, computer science, and related fields is diminishing for U.S. citizens. No undergraduate software engineering programs exist. Current computer science majors receive little exposure to software engineering principles or practices.

Initiatives: Four education initiatives are needed to address this problem:

1. Develop and deploy well-funded, high-quality education programs in collaboration with industry to entice junior and senior high school students in the U.S. to choose and prepare for careers in engineering, mathematics, and physical sciences.
2. Support development of undergraduate education curricula in software and systems engineering.

3. Create and publicize a large scholarship program to support participation by U.S. citizens in undergraduate education programs in engineering, mathematics, and science.
4. Collaborate with industry and co-sponsor a large-scale cooperative education or extended internship program for undergraduate students majoring in mathematics, engineering, and science to gain first-hand experience in research and development and applied experience in MCCR efforts. A condition for participation in this program might be a commitment on the part of students to work on MCCR efforts for a defined period after completion of a degree.

A comprehensive national education initiative akin to the National Defense Education Act (NDEA) enacted in the post-Sputnik era may be needed. It is premature for us to make such a broad and strong recommendation based on the data available for the near-term study. This issue should receive additional consideration.

Federal Policy/Practices Assessment

Recommendations for Air Force actions to improve both the contracting conditions and requirements specification activities in MCCR software-intensive systems acquisition are addressed in other studies. However, one policy and one practice we believe deserve special attention are noted here, because they may be adversely affecting the nation's MCCR software capacity.

Problem: Acquisition support for the services often is handled by a large number of contract support organizations. The size of these organizations and the roles they play in requirements specification and project performance monitoring are not well documented. If they are a drain on the labor pool of experienced engineers, they may be contributing to the software capacity problem. Since the DoD is very dependent on this set of largely unstudied organizations, it appears that DoD may be exacerbating some software capacity problems, because of inadequate information.

Initiative: Support a rigorous study of the demographics, mission, role, and impact of contract support organizations on the nation's software capacity. Use the study results to inform future policies about organic resources versus contractor support organization involvement in the software acquisition arena.

Problem: The time and cost required to gain security clearances, especially compartmented or special clearances for systems and software engineers, is substantial (from three months to one year from project inception and about $100,000 per employee).

Initiative: Commission an assessment of the current policy and practices with particular attention to provision of formal, routine procedures to prioritize processing of clearance cases. Measure the trade-offs in stringency of the current clearance allocations versus schedule slippage and cost levels resulting from current practices.

Acknowledgements

This report was made possible through the assistance of numerous individuals. In the interest of ensuring confidentiality for the participating organizations, many names are omitted. We are grateful to the senior management and technical representatives from SEI affiliate corporations who provided sensitive and crucial information to the study team. Sincere thanks are due to several MITRE professionals, especially Dick Sylvester and Jackie Kasputys, for their support and provision of data on software demand and for assistance with the survey of executives. Numerous Air Force officials from Headquarters, Military Personnel Center, Human Resource Laboratory, and ESD provided insights and data about Air Force software capacity. We are grateful also to Army CECOM officials and Navy Information Management and Laboratory officials, especially those at China Lake, who supported this effort. Finally, we want to thank the SEI resident affiliates; Members of the Technical Staff, especially John Foreman, Bill Hefley, David McKeehan, Cecil Martin, and Bob Park; Graduate Research Assistants Cap Smith, Christopher Hetrick, Jon Marom, and Tom Donovan; and Eileen Forrester, Deborah Hardy, and Maya Sternberg for their assistance in the preparation of this document.

References

[Ada 89] Ada Information Clearinghouse.
Ada Usage Database.
Ada Information Clearinghouse Newsletter.
July, 1989

[AFSCP 86] Department of the Air Force.
Air Force Systems Command Software Management Indicators.
AFSC Pamphlet 800-43.
January, 1986

[AFSCP 90] Department of the Air Force.
Air Force Systems Command.
AFSC Pamphlet 800-51.
April, 1990

[Beam 89] Beam, W. R., et al.
Adapting Software Development Policies to Modern Technology.
Technical Report, National Research Council; National Academy Press, January, 1989.

[Bluestone 89] Bluestone, I., Montgomery, R., and Owen, J.
Labor Market Opportunities for Middle Aged & Older Workers.
The Aging of the American Work Force: Problems, Programs, Policies.
Wayne State University Press, Detroit, MI, 1989.

[Boar 84] Boar, B.
Application Prototyping: A Requirements Definition Strategy for the 80s.
Wiley-Interscience, New York, NY, 1984.

[Brooks 87] Brooks, F. P., Jr.
No Silver Bullet: Essence and Accidents of Software Engineering.
COMPUTER , April, 1987.

[BrooksReport 87] Brooks, F. P., Jr.
Report of the Defense Science Board Task Force on Military Software.
Technical Report, Department of Defense, September, 1987.

[Cheney 89] Cheney, D., Secretary of Defense.
Defense Management: Report to the President.
Technical Report, Department of Defense, July, 1989.

[Chmiola 89] Chmiola, S., Lt. Col.
Software Managers - An Overview.
Briefing Slides.
March, 1989

[Coles 85] Coles, R. J., et al.
Software Reporting Metrics.
Technical Report MTR9650, Revision 2, MITRE, November, 1985.

[Curtis 88]	Curtis, B., Krasner, H., and Iscoe, N. A Field Study of the Software Design Process for Large Systems. *Communications of the ACM* 31(11):1268-1287, November, 1988.
[Cyert 63]	Cyert, R. M. and March, J. C. *A Behavioral Theory of the Firm.* Prentice-Hall, Englewood, NJ, 1963.
[DoD 88]	Department of Defense. *Software Maintenance in DoD*. Defense Analysis and Studies Office. 1988
[Firesmith 88]	Firesmith, D. G. Mixing Apples and Oranges or what is an Ada Line of Code Anyway. *Ada Letters* :110-112, September/October, 1988.
[GAO 79]	General Accounting Office. *Contracting for Computer Software Development -- Serious Problems Require Management Attention to Avoid Wasting Additional Millions*. GAO. November, 1979
[GAO 89]	General Accounting Office. *Space Defense: Management and Technical Problems Delay Operations Center Acquisition*. GAO. April, 1989
[Gries 87]	Gries, D. The 1985-1986 Taulbee Survey. *Communications of the ACM* 30(8), August, 1987.
[Gries 89]	Gries, D. and Marsh, D. The 1987-1988 Taulbee Survey. *Communications of the ACM* 32(10):1220, October, 1989.
[Hefley 88]	Hefley, W. E., et al. *Software Size and Its Impact on PDSS Needs*. Technical Report CMU/SEI-88-SR-12, Software Engineering Institute, October, 1988.
[Hughes 88]	Hughes, D. Next-Generation Defense Programs Will Increase Use of Ada Language. *Aviation Week & Space Technology* , March, 1988.
[ISSI 89]	ISSI/Lockheed. *An Analysis of Existing Software Development Practices - Process Models and Methods, Task 1 Report*. International Software Systems Inc. and Lockheed Software Technology Center. January, 1989

[Kang 89a] Kang, K. C. and Levy, L. S.
Software Reuse: What's Behind the Buzzword.
Working Papers-Unpublished.
1989

[Kang 89b] Kang, K. C. and Levy, L. S.
Software Methodology in the Harsh Light of Economics.
Working Papers-Unpublished.
1989

[Krasner 87] Krasner, H., Curtis, B., and Iscoe, N.
Communication Breakdowns and Boundary Spanning Activities on Large Programming Projects.
Empirical Studies of Programmers: Second Workshop.
1987

[Millburn 89] Millburn, George P.
Recruitment and Retention of DoD Scientists and Engineers.
Briefing Slides.
August, 1989

[Myers 88] Myers, W.
Large Ada Projects Show Productivity Gains.
IEEE , November, 1988.

[NRC 86] National Research Council.
The Impact of Defense Spending on Non-Defense Engineering Labor Markets: A Report to the National Academy of Engineering.
National Academy Press.
1986

[NSF 80] Cobern, M. and Conlon, N.
Occupational Mobility of Scientists & Engineers.
Technical Report 80-317, National Science Foundation, 1980.

[NSF 85] National Science Foundation.
Participation of Foreign Citizens in U.S. Science and Engineering.
NSF.
January, 1985

[NSF 88] National Science Foundation.
Profiles - Computer Science: Human Resources and Funding.
NSF.
November, 1988

[Personnel 88] U.S. Office of Personnel Management.
Turnover in the Navy Demonstration Laboratories, 1980-85: Management Report XI.
U.S. Office of Personnel Management.
December, 1988

[PropDat1 88]	Proprietary Data. Set 1. Unpublished. 1988
[PropDat2 89]	Proprietary Data. Set 2. Unpublished. 1989
[PropDat3 89]	Proprietary Data. Set 3. Unpublished. 1989
[Ralston 76]	Ralston, A. and Meek, C. L. *Encyclopaedia of Computer Science.* Van Nostrant Reinhold, New York, NY, 1976.
[Schultz 88]	Schultz, H. P. *Software Management Metrics*. Technical Report ESD-TR-88-001, M88-1, MITRE, May, 1988.
[Singhal 90]	Singhal, A., Rogers, E. M., Sawhney, H., and Gibson, D. *Bangalore: India's Emerging Technopolis.* Sage Publications (In Press), Beverly Hills, CA, 1990.
[White 89]	White, W. J., Maj., USAFR. Personal Correspondence. Bureau of Labor Statistics. June, 1989

Appendix A: Secondary Data Sources

Government Documents

1. Acquisition Improvement Review, Mission Critical Defense Systems, Army Materiel Command, August 31, 1988.
2. Army Science Board, 1983 Summer Study on Acquiring Army Software, Department of the Army, December 1983.
3. Boyes, Jon L., Vice Admiral, United States Navy (Retired), Armed Forces Communications and Electronics Association (AFCEA), Command & Control (C^2) System Acquisition Study, November 8, 1982.
4. Briefing Slides, Air Force Broad Area Review Briefing, Undated.
5. Briefing Slides, Air Force, Proprietary Information on Officer Retention, Undated.
6. Briefing Slides, Carter, Garry. Software Management Broad Area Review, Civilian Work Force Demographics, Air Force Civilian Personnel Management Center, August 4, 1989.
7. Briefing Slides, NAVAIR Mission, NAVAIR Systems Engineering, Undated.
8. Briefing Slides, Hefley, William. Software Engineering Institute-Definitions Effort, Undated.
9. Briefing Slides, Martin, Richard. Software Engineering Institute-Software Issues for Executives Military Computing Conference, June 6, 1989.
10. CG-47 Ticonderoga Class Guided Missile Cruiser Aegis Combat System: A Marketing Guide for Minority Business Enterprise, Latin American Manufacturers Association, Washington DC. Prepared for Minority Business Development Agency, January 1986.
11. Campi, Anthony V., Director R&D Center CECOM, Reducing Software Costs: A Corporate View, July 27, 1988.
12. Changing Perspectives for Software Development, Air Force Systems Command, Software Management Initiatives, Implementation Plan, June 23, 1989.
13. Computer Resource Management Questionnaire AFSC Systems Acquisition School, Brooks AFB, TX, Undated.
14. DACS Data Compendium, March 1986, Data Analysis Center for Software, RADC, Griffiss AFB, NY.
15. Department of the Air Force, HQ Aeronautical Systems Division (AFSC) ASD Pamphlet 800-5, Acquisition Management Software Development Capability/Capacity Review September 10, 1987.
16. Department of the Navy Directory.
17. Electronic Systems Division, Air Force Systems Command, Program Control Handbook, ESDP 600-1, 4th Edition, Volume I: The Systems Acquisition Environment, August 1982. Volume II: Financial Management, January 1983. Volume III: Scheduling, February 1983. Volume IV: Cost Estimating & Analysis, February 1984.

18. Embedded Computing Technology Office (ECTO), Naval Weapons Center, China Lake, CA, May 1988.
19. Electronic Systems Division Fall Program List, 1988 and 1989.
20. Federal Procurement Data System Standard Report, U.S. General Services Administration Federal Procurement Data Center, Office of Management and Budget, October 1987 through June 1988.
21. Federal Procurement Data System Product and Service Codes, October 1988.
22. Guidance to Electronic Systems Division Organizations in Preparing Cost Instructions for Sole/Single Source Actions and Source Selections not Based on Adequate Price Competition, October 1988.
23. Hess, James A., et al., *Report of the Army Materiel Command Software Task Force*, February 28, 1989.
24. Kubiak, Sue, Col., Chief, Software Management Group, Department of the Air Force, Software Life Cycle and People Definitions, April 14, 1989.
25. Management of Software Mission-Critical Computer Resources, Project No. 7ID-077, Intelligence, Communications and Related Programs Division, December 1988.
26. Military Standard Defense System Software Development DoD-STD-2167A, February 29, 1988.
27. Navy Fact File, 7th Edition, Office of Information, Department of the Navy, Undated.
28. Occupational Outlook Handbook 1988-89 Edition, U.S. Department of Labor, Bureau of Labor Statistics.
29. Personnel Demonstration Project, Naval Weapons Center, China Lake, CA, Undated.
30. Quarterly Officer Retention Report, HQ Air Force Military Personnel Center, United States Air Force, Randolph AFB, June 30, 1989.
31. Reed, L. S., Col., Life Cycle Software Support (LCSS), Mini Functional Area Assessment, Part III-B, Undated.
32. *Report of the Defense Science Board Task Force on Military Software*, Office of the Under Secretary of Defense for Acquisition, Washington DC, September 1987.
33. Shumskas, Anthony F., Maj., Department of the Air Force, HQ, Air Force Systems Command, AFSC Pamphlet (AFSCP) 800-43, February 19, 1986.
34. Washington Information Directory, Congressional Quarterly Inc., 1988-89.

Non-Government Documents

1. 1988/89 Professional and Scientific Personnel Report (Geographic Edition), Volume I, Executive Compensation Services Co., July 1988.
2. *Abbreviations, Acronyms & Directives*, Signal, February 1989.
3. Ardis, M., et al., Technical Report CMU/SEI-89-TR-21, *1989 SEI Report on Graduate Software Engineering Education*, June 1989.

4. ASD Project List, Air Force Magazine, January 1988.
5. Boehm, Barry W., *Improving Software Productivity*, Twenty-Third IEEE Computer Society International Conference, IEEE Catalog no. 81ch1702-0, September 1981.
6. Camp, Ray A., et al., 1987 ASCQ Quality Congress Transactions-Minneapolis, *Why Johnny Can't Program*, Princeton University.
7. Canan, James W., *How Electronic Counter Measures Went Wrong*, Air Force Magazine, August 1989.
8. Carpenter, Maribeth, CMU/SEI-89, *Continuing Education in Software Engineering: A Report on a Prerequisite Planning Workshop*, September 1989.
9. Computer Science Department List of Students Completing the PhD by August 1985, Carnegie Mellon University.
10. Computer Science Department List of Students Who Expect to Receive their Doctorate Degree - 1985, Massachusetts Institute of Technology.
11. Foreman, John and Goodenough, John, Technical Report CMU/SEI-87-TR-9, *Ada Adoption Handbook: A Program Manager's Guide*, May 1987.
12. Frenkel, Karen, *Toward Automating the Software-Development Cycle*, Communications of the ACM, Volume 28, Number 6, June 1985.
13. Hansen, Greg et al., Technical Report CMU/SEI-87-TR-12, *The Analysis of the Technical Order Production Process at Ogden Air Logistics Center and Recommendations for the Improvement of the Process*, January 1988.
14. Krasner, Herb, et al., *A Methodology for Studying Software Design Teams: an Investigation of Conflict Behaviors in the Requirements Definition Phase*, Empirical Studies of Programmers: Second Workshop, 1987, pp.83-99.
15. Parnas, David L., Education for Computing Professionals, Technical Report 89-247, Department of Computing & Information Science, Queen's University, Kingston, Canada (ISSN-0836-0227), March 1989.
16. *Topical Guide to AFCEA Member Companies*, Signal, February 1989.
17. Tulic, Martin W., AT&T Bell Laboratories, Final Report of the Software Engineering Curriculum Planning Committee - Work Project No. 9655990001, October 21, 1988.
18. Vetter, Betty M., The Technological Marketplace Supply and Demand for Scientists and Engineers, Scientific Manpower Commission, May 1985.
19. Visser, Willemien, *Strategies in Programming Programmable Controllers: A Field Study on a Professional Programmer*, Empirical Studies of Programmers: Second Workshop, 1987, pp.217-30.
20. Wise, Lauress L., American Institutes for Research, *Project A: Findings and Futures*, November 12, 1982.
21. Zavala, A., The Human Factors of Programmer Productivity, Lockheed Missiles and Space Company inc., April 4, 1988.

Appendix B: Near-Term Study Questionnaires

Executive Questionnaire

Sample Interview Questions

Human Resources/Personnel Interview Guide

Table of Contents

List of Figures

List of Tables