Department of Defense Software Factbook

Bradford Clark
Christopher Miller
James McCurley
David Zubrow
Rhonda Brown
Mike Zuccher

April 2017

SPECIAL REPORT
CMU/SEI-2017-TR-004

Software Solutions Division

http://www.sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

Table of Contents

List of Figures

List of Tables

Abstract

This Department of Defense (DoD) Software Factbook provides an analysis of the most extensive collection of software engineering data owned and maintained by the DoD, the software resources data report (SRDR). The SRDR is the primary source of data on software projects and their performance.

The Software Engineering Institute analyzed the SRDR data and translated it into information that is frequently sought-after across the DoD. Basic facts are provided about software projects, such as averages, ranges, and heuristics for requirements, size, effort, and duration. Factual, quantitatively derived statements provide easily digestible and usable benchmarks.

Findings are also presented by system type or super domain. The analysis in this area focuses on identifying the most and least expensive projects and the best and worst projects within three super domains: real time, engineering, and automated information systems. It also provides insight into the differences between system domains and contains domain-specific heuristics.

Finally, correlations are explored among requirements, size, duration, and effort and the strongest models for predicting change are described. The goal of this work was to determine how well the data could be used to answer common questions related to planning or replanning software projects.

1 How to Read this Document

This Department of Defense Software Factbook is an analysis of the most extensive collection of software engineering data owned and maintained by the DoD. It explores the contents of the data set and provides high-level, DoD-wide heuristics as well as domain-specific benchmark data. Each section is described below to help you locate the facts most applicable to your situation and needs.

Executive Summary

The Executive Summary contains the highest level summary of analysis results and provides general answers to commonly asked questions. It provides frequently sought-after information and heuristics that can establish much needed context about software development across the DoD.

DoD Software Project 101 – Basic Facts

The Basic Facts section provides averages, ranges, and heuristics via descriptive statistics of the key software parameters (requirements, ESLOC, effort, and duration/schedule). The analysis is translated into factual, quantitatively derived statements to provide easily digestible and usable benchmarks. For example: Based on the 198 real-time projects analyzed, a typical real-time build project consists of 449 requirements and 35,000 ESLOC, requires about 40,000 person hours with a staff of 8 people, and takes about 3 three years to complete.

Portfolio Performance

This section highlights findings by system type, or super domain. The analysis focuses on identifying the most and least expensive projects, as well as the best and worst projects within three super domains: real time (RT), engineering (ENG), and automated information systems (AIS). It also provides insight into the differences between system domains and contains domain-specific heuristics.

Project Planning, Trade-offs, and Risk

In this section, we present our findings from a more extensive analysis of the data, where we explored correlations among requirements, size, duration, and effort. The goal of this work was to determine how well the data could be used to answer common questions related to planning or replanning software projects, such as “How much growth should we plan for?” and “How well can initial estimates be used to predict final outcomes?”

Although more analysis will be done in this area as we obtain new data, we present the strongest models we found to predict changes in factors such as requirements, schedule, and productivity.

2 Executive Summary

This Factbook presents an analysis of software engineering data gathered by the DoD from Software Resources Data Reports (SRDRs).[1] The conclusions and benchmarks are statistically derived from real projects from the SRDR database; therefore they can be traced back to the source. Given the compilation across system domains, development organizations, and languages, this data summary is most useful to high-level decision makers. The data can be used as a general rule of thumb when discussing software as part of the system at large, and the numbers we provide allow program managers and other senior engineering staff to answer common questions from senior executives about DoD software projects in general.

Understanding Typical DoD Projects

The table below presents the highest level summary of our analysis results to answer commonly asked questions about typical software projects. These heuristics are intended for those who simply want a general idea of how much a software project might cost or how long it might take. Results from the 25th and 75th percentiles are also provided along with the average or typical result to make it easier for you to compare your project to other DoD projects in the “normal” range.

DoD Software Projects: Basic Benchmarks	*Small projects (25th percentile)*	***Average/Typical***	*Large projects (75th percentile)*
Requirements: What is the functional size of a DoD software project?	100 requirements	**400 requirements**	1100 requirements
ESLOC: How many lines of code do DoD software projects contain?	12,000 lines of code	**40,000 lines of code**	110,000 lines of code
Effort: How many hours of work does it take to complete DoD software projects?	13,000 hours	**40,000 hours**	97,000 hours
Duration: How long do DoD software projects last?	22 months	**35 months**	48.3 months
Team size: How many people work on DoD software project teams?	3.1 FTEs	**8 FTEs**	19.4 FTEs
Productivity: How many lines of code per hour do DoD software projects produce?	0.56 ESLOC per hour	**1.07 ESLOC per hour**	1.69 ESLOC per hour
Cost: How much do DoD software projects cost?*	$1.1 M	**$3.3 M**	$8 M

**Based on an $82.24 hourly rate*

The data set for this analysis used 287 projects from DoD SRDRs submitted by contractors for MDAP and MAIS projects.

[1] For a full explanation of the data analyzed, see Appendix K: Data Source Details.

Notable Conclusions by Super Domain

Beyond these basic benchmarks, findings in this report are also presented by system type or super domain. Further correlations are then explored among requirements, size, duration, and effort. Some of the most notable conclusions from our analyses are described below.

Software growth can be predicted from initial estimates.

Initial estimates enable statistically strong predictions of the realized software requirements, size, effort, and schedule reported upon final delivery. Schedule duration can also be predicted separately for Army, Air Force, and Navy programs. Predictions of productivity (ESLOC/person-month) are of moderate strength but can also be calculated separately for three super domains (automated information systems, engineering, and real time). Productivity (ESLOC/person-month) predictions would dramatically strengthen from a mid-course or interim data report.

Real-time software is the most expensive software to develop, followed by engineering and automated information system software.

The software data were divided into three super domains for analysis: real-time, engineering, and automated information system software.[2] Analysis revealed that real-time software costs 14% more to develop than engineering software, and 39% more than automated information system software. The average cost per day for an average-size project is $3,324 for real-time, $2,912 for engineering, and $2,393 for automated information systems.

Best-in-class software projects show significant gains in efficiency, speed, and cost reduction.

Each group of software data was analyzed for best- and worst-in-class performance using an average-size project. Performance is defined in terms of development unit cost (efficiency), production rate (speed), and total cost.

Analysis showed that best-in-class real-time projects are 2 times more efficient than average projects and 4.7 times more efficient than worst-in-class projects. Best-in-class projects are also 1.8 times faster than an average project and 3.4 times faster than a worst-in-class project. Best-in-class projects cost $1.510M and worst-in-class projects cost $7.047M.

Best-in-class engineering projects are 2.3 times more efficient than average projects and 5.3 times more efficient than worst-in-class projects. The best-in-class project is 1.6 times faster than an average project and 2.6 times faster than a worst-in-class project. Best-in-class projects cost $1.190M and worst-in-class projects cost $5.385M.

The best-in-class automated information system projects are 1.7 times more efficient than average projects and 3 times more efficient than worst-in-class projects. Best-in-class projects are 2 times faster than average projects and 4 times faster than worst-in-class projects. Best-in-class projects cost $1.842M and worst-in-class projects cost $5.62 M.

[2] See Appendix C for a comprehensive description of the super domains.

3 Introduction: DoD Software Projects 101 – Basic Facts

This Factbook provides an analysis of the most extensive collection of software engineering data owned and maintained by the DoD, the software resources data report (SRDR).[3] The SRDR is a contract data deliverable that formalized the reporting of software metrics data and is the primary source of data on software projects and their performance. The SRDR reports are provided at the project level or subsystem level, not at the DoD Acquisition Program level. The data points analyzed in this report are representative of software builds, increments, or releases. In many cases, several data points from the same Program are contained in the data set.

The SRDR applies to all major contracts and subcontracts, regardless of contract type, for contractors developing or producing software elements within acquisition category (ACAT) I and IA programs and pre-MDAP and pre-MAIS programs subsequent to milestone A approval for any software development element with a projected software effort greater than $20M.[4]

It is designed to record both the estimates and actual results of new software developments or upgrades. The majority of the SRDR data used in this analysis is based on the final report that contains actual result data. Data for this analysis had to include the following information:

- size data (functional and product)
- effort data
- schedule data

The data set we used for this analysis included **287 projects** from the product-event final report data. Similarly, we used **181 pairs** of initial and final cases for analysis of the estimated versus actual performance in Section 2.

3.1 Key Project Dimensions and Empirical Relationships

Since the 1970's research has been conducted into how to estimate to cost of software development. An entire industry focused on parametric software estimation has grown, and at the core of this industry is a fundamental assumption that the cost of developing software can be estimated based on an accurate estimate of the size of the software product to be developed. This concept might be more accurately described as an assumed empirical relationship between cost and software size.

Figure 1 shows key parameters related to software cost: functional size (in requirements), physical size (in equivalent source lines of code), effort hours, and duration of software projects. In most DoD environments size is measured by requirements and the final physical size of the software product, which is commonly measured in source lines of code. The amount of effort required to deliver the software can be estimated if you know the size. Similarly, duration (or schedule) can be derived from the effort.

3 For a full explanation of the data analyzed, see Appendix K: Data Source Details.

4 CSDR Requirements, OSD Defense Cost and Resource Center, http://dcarc.cape.osd.mil/CSDR/CSDROverview.aspx#Introduction

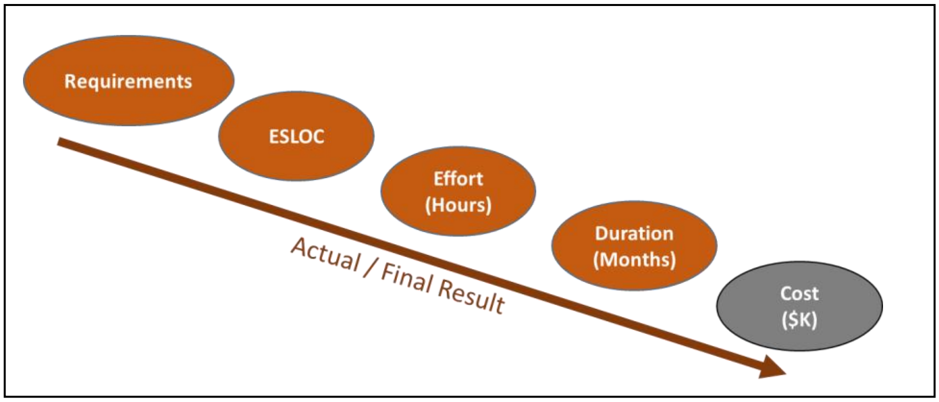


Figure 1: Key Software Parameters

Using the SRDR data for 287 data sets, each of the four key parameters is statistically described in Section 3.2 through Section 3.5. Section 3.6 looks at typical team size, Section 3.7 examines productivity, and Section 3.8 combines the results into a statistical view of a typical DoD software projects.

Defining "Typical" in DoD Software Projects

The number most people seek when asking about the analysis of software data is the average. When someone asks, "What is the average size/cost/duration of a software project?" they are looking for a general idea of the most common or *typical* result. It is rare for a program manager or other senior executive to ask for the statistically derived average, which is influenced by extreme values in the data set. Our use of the word "average" in this report follows common use and does not, in general, refer to the statistical concept.

When the data set is normally distributed, the average provides a sound measure of the center of the data. The challenge is that our key software project parameters are not normally distributed (see Figure 2). The red line in the figure shows the distribution of the size data is skewed to the left, up against zero. Therefore, we normalized the data by transforming it by its natural log. Both the raw descriptive statistics and the natural logarithmic statistics were used to draw conclusions. Each of the analyses in this section provides an average project parameter in the general sense to be used as a heuristic to assist decision makers.

3.2 Functional Size (Requirements)

Functional size represents the overall magnitude of the software capabilities without regard to the final solution. The benefit of using functional measures is their availability early in the software development lifecycle. In the DoD acquisition community, requirements are rigorously derived and used as the contractual basis for acquiring systems. Therefore requirements and requirements documents are produced as part of the system acquisition life cycle and are readily available for the extraction of the number of requirements.

The drawback of using functional measures is that the requirement does not consistently correlate to a unit of effort (i.e., not all requirements take the same amount of effort to satisfy). Using the total number of requirements to represent size is useful, but trying to attach a unit cost (i.e., the cost per requirement) is not advised.

Figure 2 shows the skewed nature of the raw data related to requirements. The bulk of the data lies between 102 (~100) and 1110 (~1100) requirements, which is a large range. Once the data is normalized using a natural log transformation (shown in Figure 3), the median is $e^{6.04}$, or 420 requirements with a mean of 368 requirements. Both are much closer to the raw data median of 399 than the raw data mean of 1118 requirements.

Requirements data analyzed by super domain are presented in Figure 4. As is in shown on the top of the figure, to the left of the line is the 25th percentile value. This indicates that 25% of the projects have less than 100 requirements. Similarly, on the right the 75th percentile value indicates that 25% of the projects have more than 1100 requirements. Note that 50% of the projects have between 100 and 1100 requirements, with relatively more toward the lower end and a median or typical view of 400. The additional lines in the figure can be similarly interpreted. Similar figures are provided throughout this section showing the 25th percentile, median, and 75th percentiles.

An easy heuristic for the average functional size of a DoD software project is **400 requirements**.

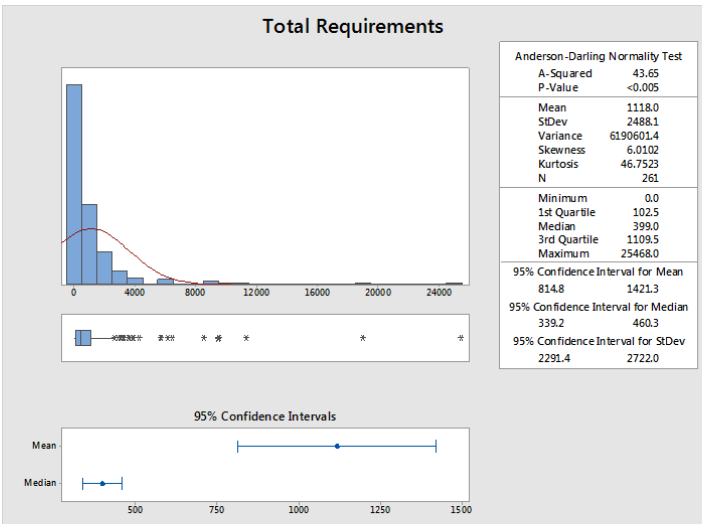


Figure 2: Functional Size

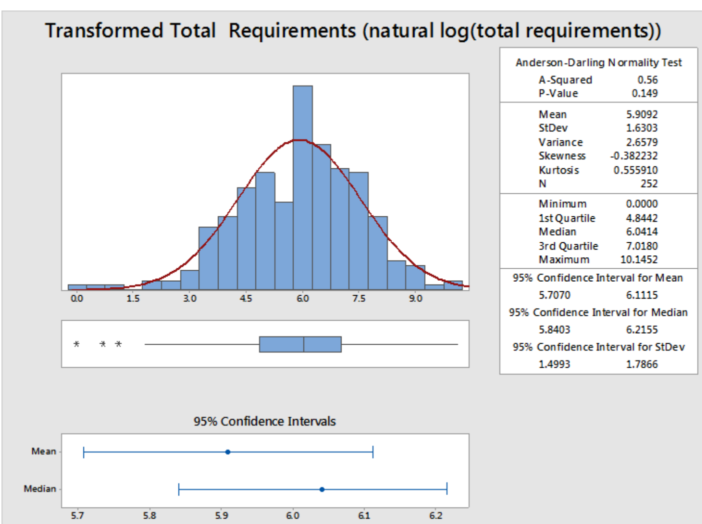


Figure 3: Functional Size, Normalized

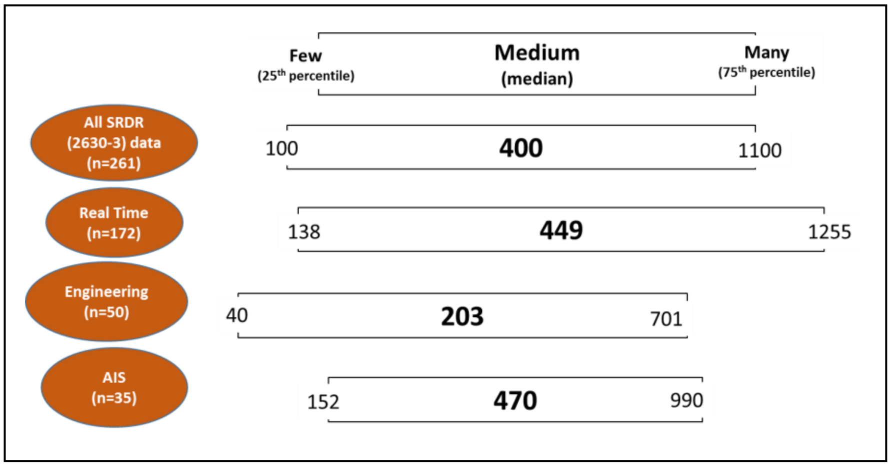


Figure 4: Requirements Data by Super Domain

3.3 Product Size (ESLOC)

Another common measure of interest is product size, which is often measured in source lines of code (SLOC). A key issue in using SLOC as a measure of work effort and duration is the difference in work required to incorporate software from different sources, including:

- new code
- modified code (changed in some way to make it suitable)
- reused code (used without changes)
- auto-generated code (created from a tool and used without change)

Each of these sources requires a different amount of work effort to incorporate into a software product. The challenge is in coming up with a single measure that includes all of the code sources. Equivalent source lines of code (ESLOC) normalize all code sources to the equivalent of a new line of code by computing a portion of the physical measures for modified, reused, and auto-generated code.[5]

Figure 5 shows the ESLOC data, and Figure 6 shows it normalized using a natural log transformation. ESLOC by super domain is presented in Figure 7. An easy heuristic to use for average project size is **around 40,000 ESLOC** for all projects.

[5] This is explained in more detail in Appendix B.

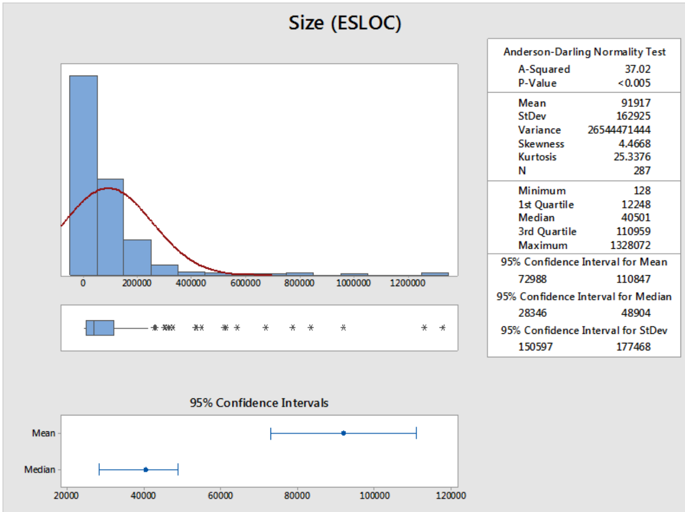


Figure 5: Product Size in ESLOC

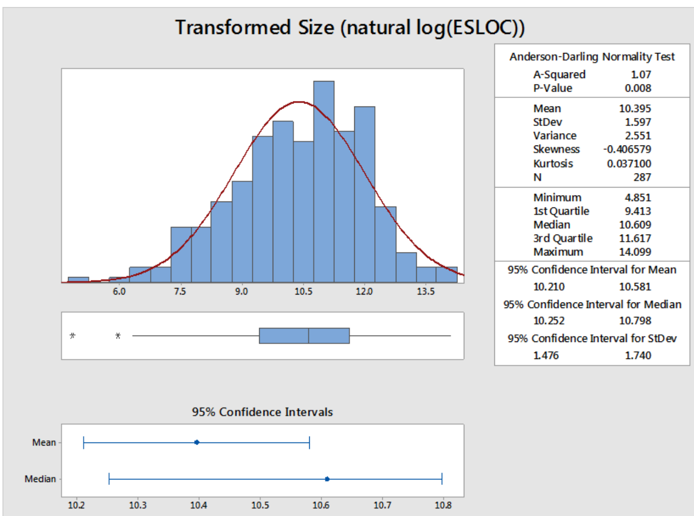


Figure 6: Product Size in ESLOC, Normalized

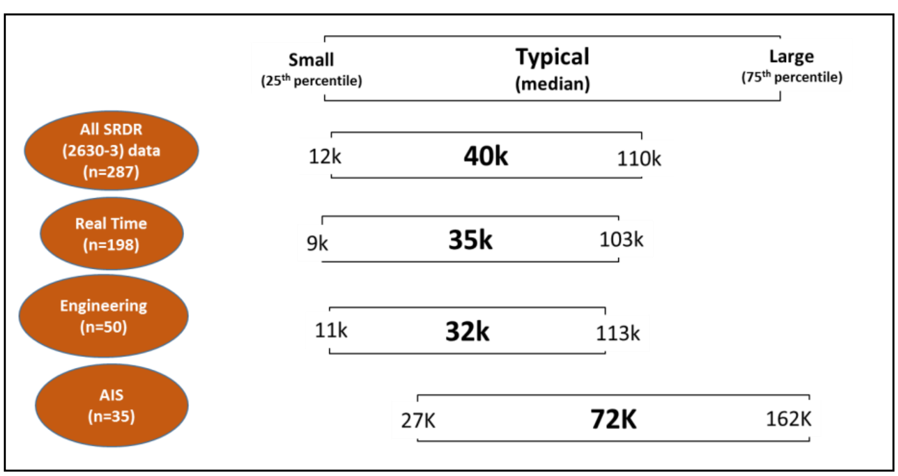


Figure 7: ESLOC by Super Domain

3.4 Effort

The amount of effort used to create software is the major driver of the cost of the development; the effort estimate in dollars provides the largest element in the cost estimate for software. Effort is usually collected in hours. For simplification purposes many estimation tools and equations use person months. When comparing effort data, ensure that the same conversion rate is used across the data set (i.e., the number of hours in a person month and/or number of hours in a full time equivalent). As detailed in Appendix G: Burden Labor Rate, it is assumed here that there are 152 hours in a labor month and 1824 hours per full-time equivalent (FTE), based on an annual labor rate of $150,000.

Figure 8 shows the effort data; Figure 9 shows that data normalized. The effort hour data analyzed by super domain are presented in Figure 10. An easy heuristic to use for average project effort is around **40,000 hours, 263 person months, or 22 FTEs** for a DoD software project.

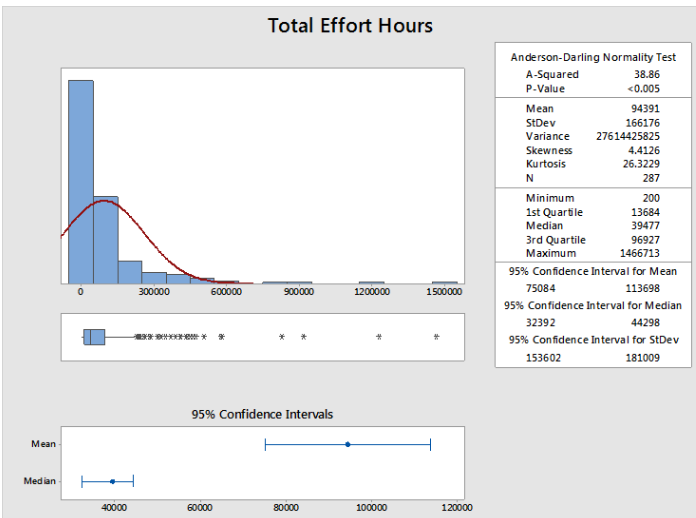


Figure 8: Effort

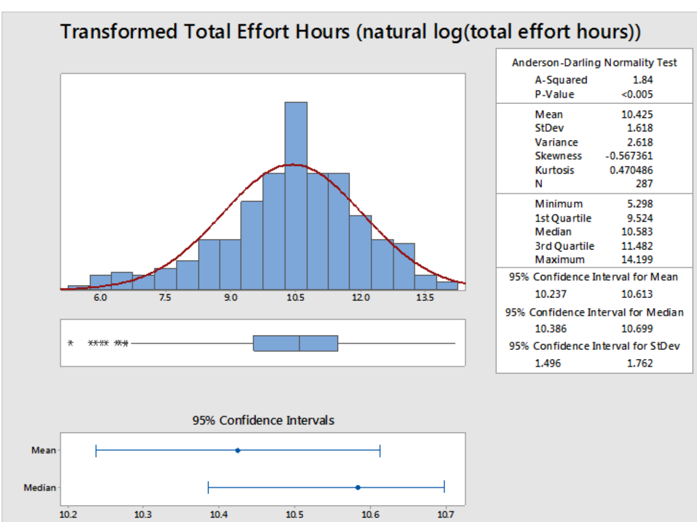


Figure 9: Effort, Normalized

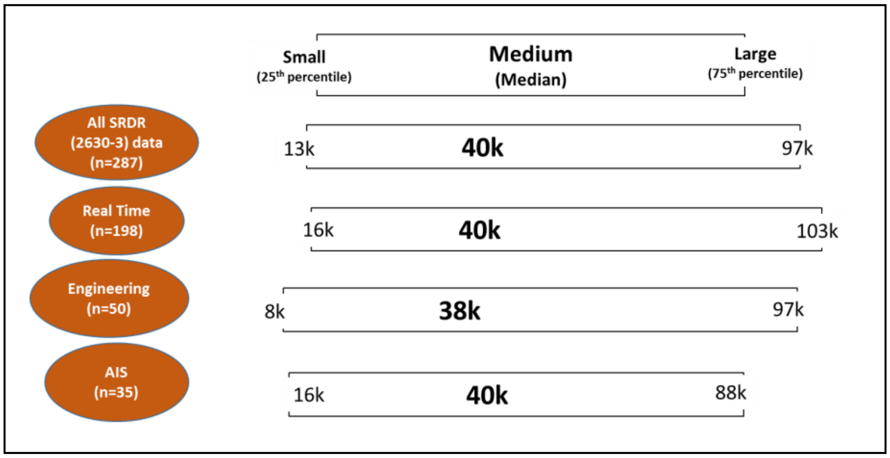


Figure 10: Effort Hours by Super Domain

3.5 Duration

Duration is a measure of the calendar time it takes to complete the software project. Many factors affect duration, including staffing profile, schedule constraints, and release plan. No adjustments are made for these factors in the data reported in this section.

Figure 11 shows that most projects have a duration between 22.0 months and 48.3 months with a median duration of 35. Figure 12 shows the data normalized. The data indicate that the majority of projects take between 2 ½ to 3 years. An easy heuristic to use for the duration of an average DoD software project is **approximately 3 years**.

Duration data analyzed by super domain is presented in Figure 13.

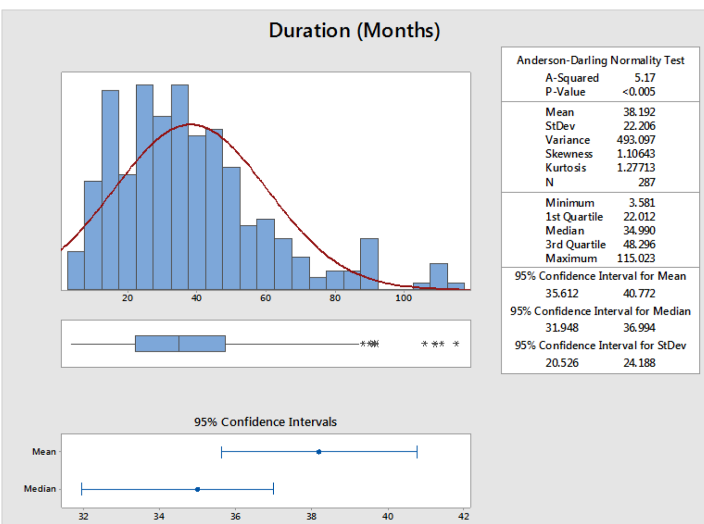


Figure 11: Duration

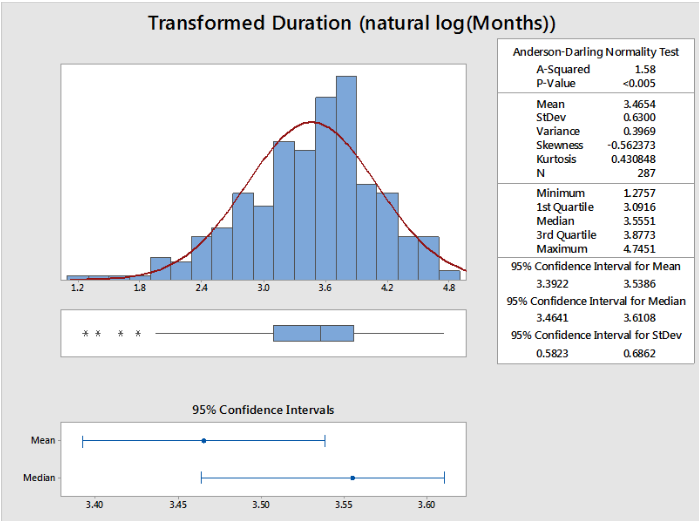


Figure 12: Duration, Normalized

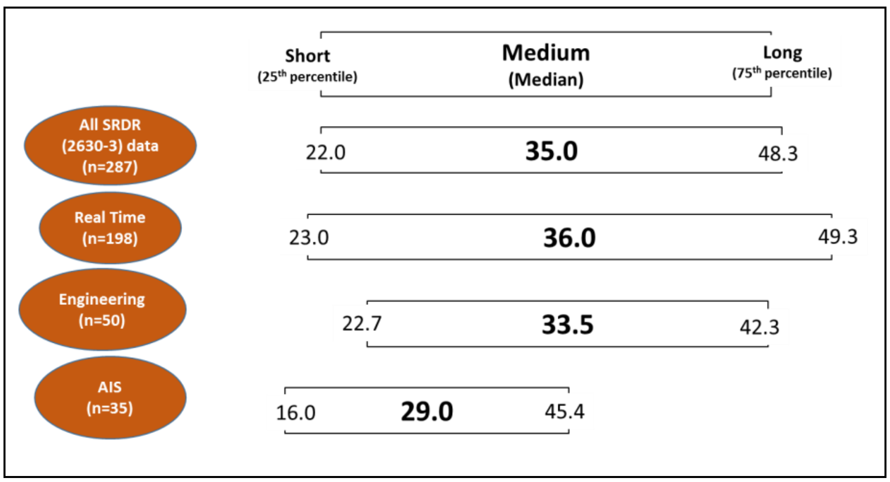


Figure 13: Duration Data by Super Domain

3.6 Team Size (People)

The size of the development team reported here is based on measures of project effort and duration. The effort for a project is reported in labor hours. Labor hours are converted to person months of effort (based on 152 hours/month) and divided by months of project duration. This derives the average level of project staffing or full time equivalent (FTE). The FTE for the 287 data points can be seen in Figure 14.

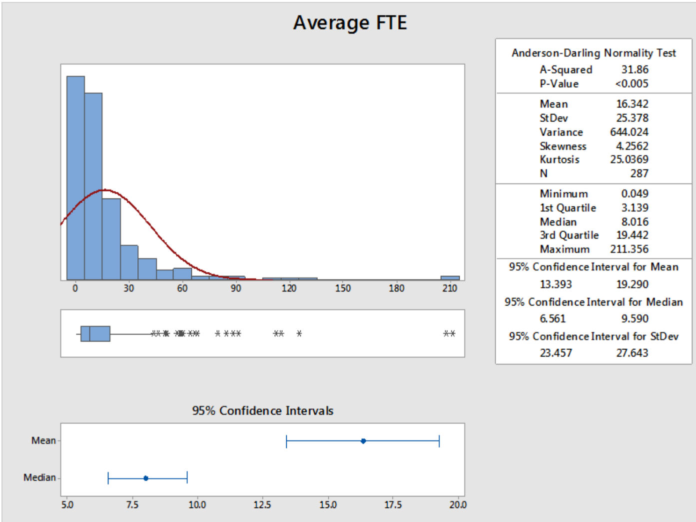


Figure 14: Team Size

Figure 15 shows a histogram of the same data in natural log scale. It shows that most teams have 20 or fewer people. Recall that SRDRs are required for contracts over $20 million. These contracts have multiple product events resulting in seemingly small team sizes which, in fact, are due to low levels of effort over relatively long durations.

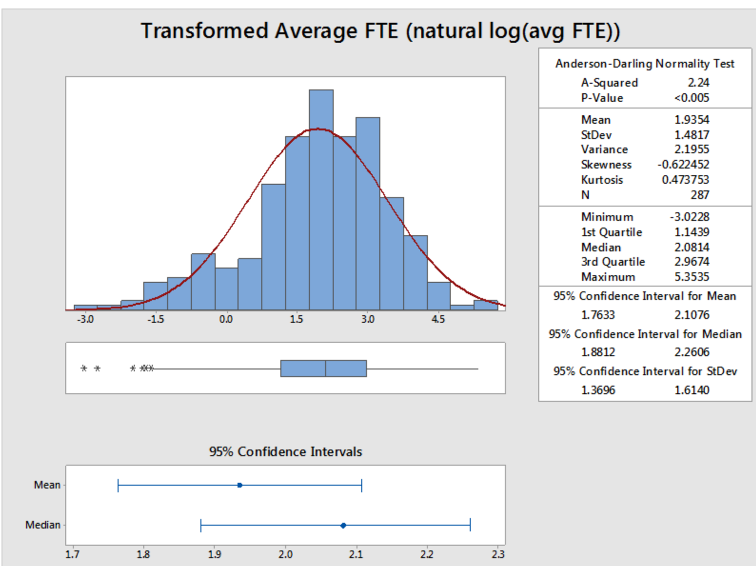


Figure 15: Time Size, Normalized

Figure 16 shows the data divided into three groups: small-, medium-, and large-team-size projects. The groups are based on a cumulative percentage divided into thirds. Small teams make up the lower third, medium size teams are in the middle third, and large teams make up the upper third. Based on the groupings the team sizes are as follows:

- small-size teams: < 5 average staff
- medium-size teams: 5-14 average staff

- large-size teams: > 14 average staff

Medium and large team sizes are used in the effort/schedule tradeoff analysis.

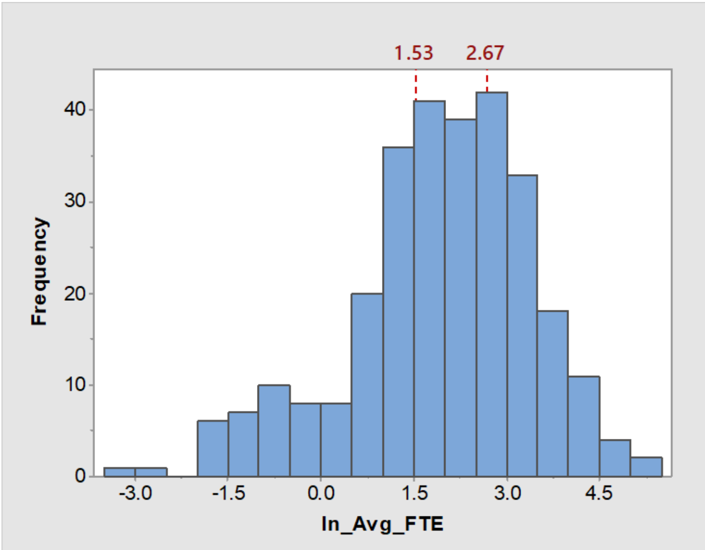


Figure 16: Team Size Distribution

Duration data analyzed by super domain is presented in Figure 17.

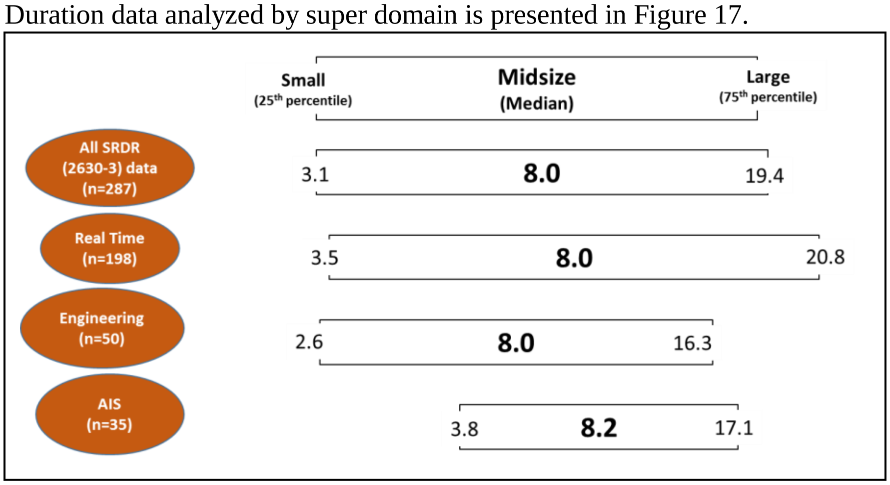


Figure 17: Duration Data by Super Domain

3.7 Productivity

Productivity (also referred to as efficiency) is the amount of product produced for an amount of resource. For software, productivity is commonly measured by size (ESLOC) divided by effort hours.

Productivity in general is considered very competition sensitive and therefore rarely shared publicly by the private sector. Since the SRDR data set is owned and maintained by the government and the individual data provider's productivity is protected, this compilation of data provides a rarely available insight into software productivity across the industrial base.

Figure 18 shows the raw productivity data, and Figure 19 shows the data after normalization. For practical purposes, the data shows a 1:1 ESLOC: hour ratio. Duration data analyzed by super domain is presented in Figure 20.

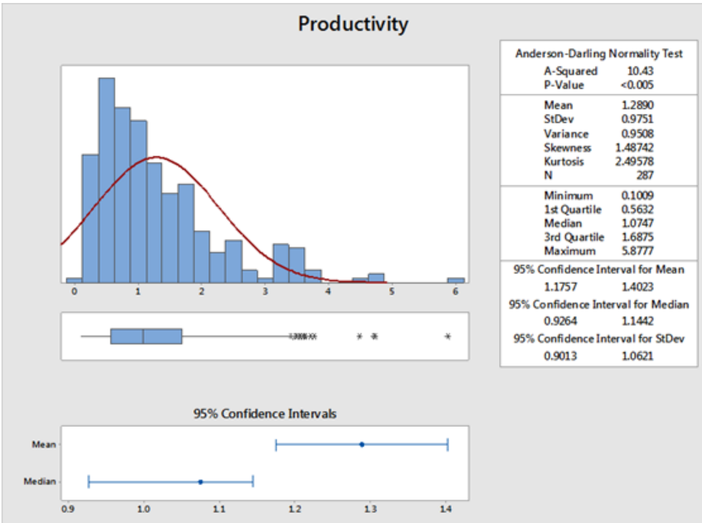


Figure 18: Productivity

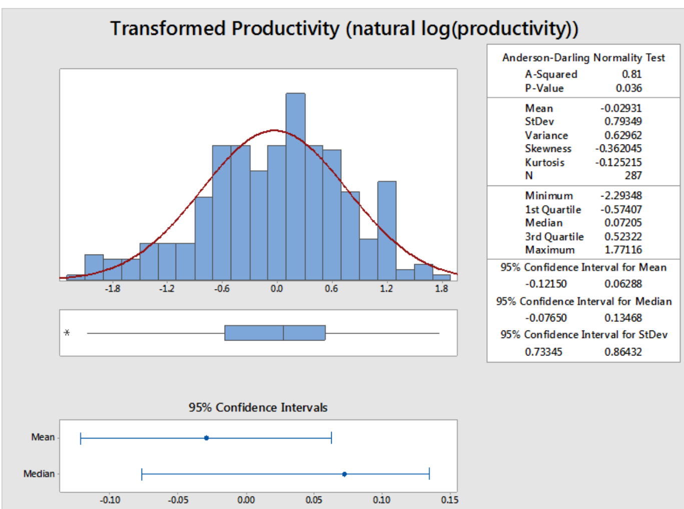


Figure 19: Productivity, Normalized

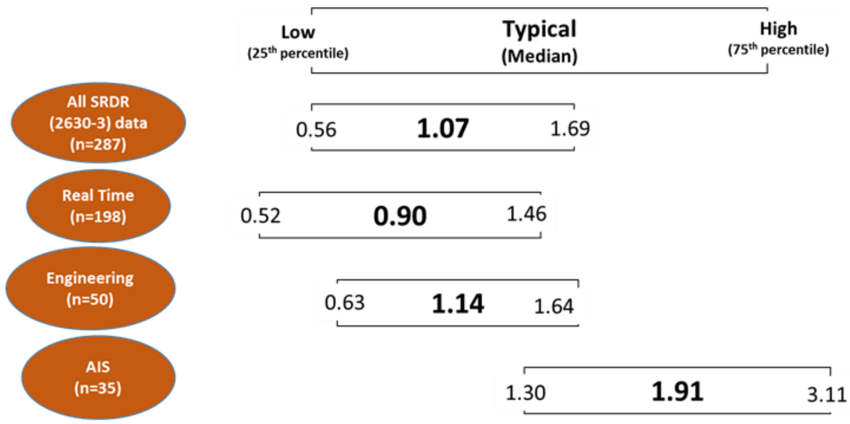


Figure 20: Duration by Super Domain

3.8 Summary: Profiles of Typical Projects

Integrating the analysis results of the individual parameters provides a general software project profile. This section contains the profiles for a generic DoD software project, as well as profiles for RT, ENG, and AIS projects.

3.8.1 Snapshot of a Typical DoD Software Project

Figure 21 provides a snapshot of the overall dataset, showing the size and scope of a typical DoD software project. Keep in mind SRDR data points are typically submitted by subsystem or potential increment; these numbers do not represent an entire DoD program of record.

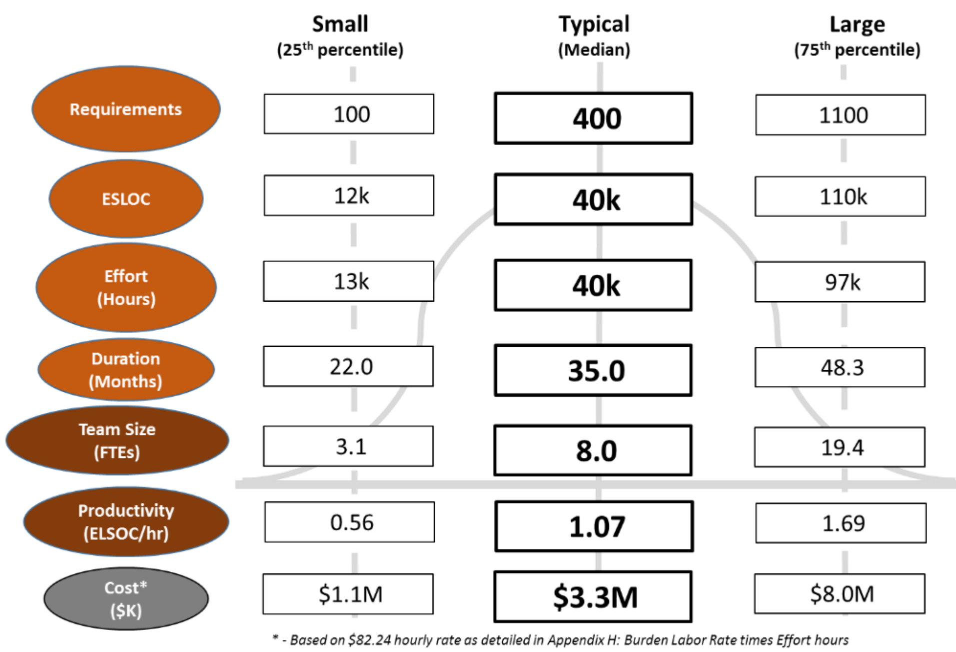


Figure 21: Parameters of DoD Software Projects

This data can be used to answer general questions about DoD software projects. For example,

- Question: What is the typical (average) size of a software delivery?
  Answer: 40 KESLOC
- Question: How long does an increment take?
  Answer: 35 months (~3 years)
- Question: How many FTEs does a typical software project require?
  Answer: 8 FTEs; some large projects may require upwards of 20 FTEs.
- Question: In general how much does a software project cost?
  Answer: Software projects tend to range between $1 and $8 M; without knowing any details about what type of software or its composition, a generic DoD project costs a little over $3 M.

The percentile numbers help convey the variation in the data. These data can be utilized by oversight offices when assessing overall program feasibility. A project plan that contains parameter values outside the 25th and 75th percentile range signifies a situation that is not common and might require additional scrutiny. In this case, the oversight office would want to ask for more information about the engineering and technical rationale to justify this plan.

Given the mix of system domains, language types, environments, platforms, functionality, and associated quality/performance parameters, these rules of thumb may not provide a lot of value to project managers estimating their software efforts. To get the information useful to them, they would need to isolate like projects in the dataset and generate a parameter profile that best represents the system they are developing. In this vein, the following sections provide heuristics by super domains.

3.8.2 Snapshot of Real Time Software Projects

RT software is typically the most complex and intricate type of software. It tends to be embedded in the system architecture and contributes to the success or failure of key performance parameters of the system. Given the level of rigor this type of software requires, the variations between the RT super domain parameters in Figure 22 are not surprising. Of the 287 data points analyzed, 198 were classified as real time.

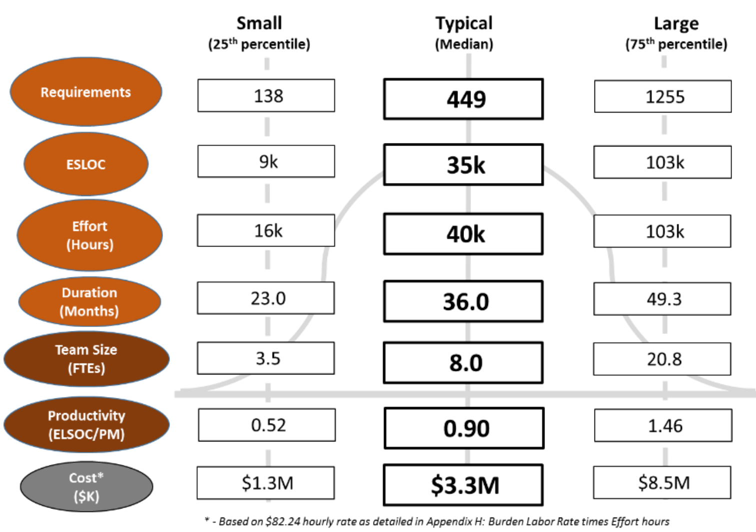


Figure 22: Parameters of Real Time Software Projects

It is logical that increased system complexity would require a more detailed articulation of the requirements, resulting in a higher requirements count and lower productivity in comparison to the overall data set. This can also be seen in the slightly higher effort hour percentile values.

3.8.3 Snapshot of Engineering Software Projects

ENG super domain software is of medium complexity. It requires engineering external system interfaces, high reliability (but not life-critical) requirements, and often involves coupling of modified software. Examples of software domains in this super-domain are: mission processing, executive, automation and process control, scientific systems, and telecommunications.

Figure 23 shows the key software parameters for the 50 ENG super domain data points in the 287 data set.

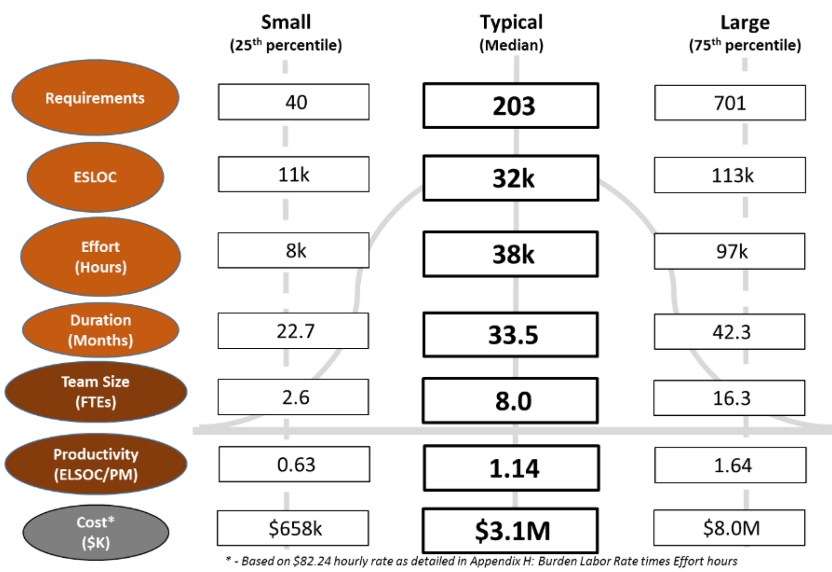


Figure 23: Parameters of Engineering Software Projects

In comparison to RT systems, ENG systems tend to state their requirements at a slightly higher level. For example, a typical requirement may be, “System X shall interface with System Y.” In this case there are several nuances to meeting this requirement. This can be seen by comparing the requirements parameters, ESLOC, and effort parameters of the RT data to the ENG data.

3.8.4 Snapshot of Automated Information System Software Projects

AIS software automates information processing. These applications allow the designated authority to exercise control over the accomplishment of the mission. Humans manage a dynamic situation and respond to user input in real time to facilitate coordination and cooperation. Examples of software domains in this super-domain include intelligence and information systems, software services, and software applications.

Figure 24 shows the key software parameters for the 35 AIS super domain data points in the 287 data set.

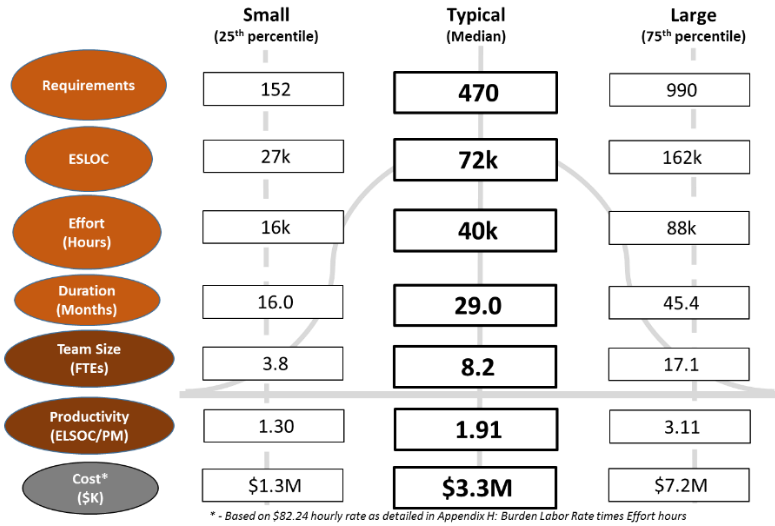


Figure 24: Parameters of Automated Information System Software Projects

The size and productivity parameters vary the most from the overall super domain parameters. Based on the way AIS are developed (i.e., adaptation of existing COTS/GOTS applications), the increase in comparison to the other super domains is not surprising.

3.8.5 Using the Heuristics

For years, the software engineering community has avoided quantifying the basic parameters of software development. Our analysis provides high-level summaries from which useful (albeit very simplified) heuristics can be established. Responsible use is encouraged. When communicating the heuristics contained in this Factbook, it is advised to caveat the data with, "It depends, but based on 287 data points from the SRDR database, a typical software project …"

4 Portfolio Performance

This section explores the findings by super domain to answer some common questions about different software types.

4.1 Most and Least Expensive Software

What are the most and least expensive software types to develop?

Our analysis is based on the rationale that some types of software are more difficult to develop than other types and therefore require more effort to develop. The level of difficulty can be caused by factors such as execution timing constraints, interoperability requirements, commercial-off-the-shelf (COTS) software product incorporation, algorithmic complexity, communication complexity, data-bandwidth requirements, and security requirements. To account for the dissimilarities in project difficulty, projects are segregated into three super domains.

The analysis proceeds by introducing two concepts: unit cost and production rate.

- Unit cost is the cost of producing a unit of software with some amount of effort. In this case, the unit of software is thousands of equivalent source lines of code (KESLOC).[6] The effort is reported in labor hours, which can be translated into cost using an average labor rate.
- Production rate is the rate at which a unit of software is delivered over a period of time. The unit of software is a KESLOC and the time is days of project duration.
- Cost is derived by applying a burdened labor rate[7] to the number of labor hours worked in a day. Hours per day are determined by dividing total hours by the duration (total days). For example, if a real time project required 1,007 total hours and 25 days, the labor hours expended in a day is 40.3 (implying several people were working on the project).

The analysis then normalizes the unit cost with the production rate, creating a high-level comparison. This is done because some projects may choose to employ more staff to increase their production rate and deliver the software sooner or vice versa. The resulting effort per day is then multiplied by an average burden labor rate to derive cost.

4.1.1 Unit Cost

With an average project size of 40,000 ESLOC, each of the three groups are analyzed separately. Trends for each group were created based on a natural log-transformation of the data. This transformation made it clearer to see the relationships between the three groups for an average project size of 40,000 ESLOC.

The difference in unit costs between the three groups is shown in Table 1. Real-time software shows that for small amounts of size, a large amount of effort is required. Automated information system software data shows the opposite: for large amounts of size, a small amount of effort is required.

---

6 The rationale and formulation of ESLOC is explained in Appendix B.

7 Burden labor rate used in this analysis is explained in Appendix G.

Table 1: Unit Costs for Different Domains

Domain	Hours / KESLOC
Real Time Software	1,070
Engineering Software	936
Automated Information System Software	578

4.1.2 Production Rate

The production rate data analysis focused on the relationships between size and duration for the three super domains. The analysis revealed much greater variability than the unit cost plot. This indicates a very weak systematic relationship between size and duration. The dispersion of the data is attributed to other factors that influence the size-duration relationship (e.g., different levels of staffing on similar size projects can impact duration). This is an area for further research.

For an average-size project, the production rate (how long it takes to deliver a unit of software) is shown in Table 2.

Table 2: Production Rate for Different Domains

Domain	Days / KESLOC
Real Time Software	25
Engineering Software	26
Automated Information System Software	20

4.1.3 Cost Comparison

When unit cost is divided by production rate, the average number of hours each month is determined. Using an average burden labor rate,[8] the normalized monthly cost for each group is shown in Table 3. The hours/day indicate that more than one person is working per day.

Table 3: Costs for Different Domains

Domain	Hrs / Day	Cost / Day
Real Time Software	40.4	$3,324
Engineering Software	35.4	$2,912
Automated Information System Software	29.1	$2,393

Real-time software is the most expensive to develop and automated information system software is the least expensive. RT software costs 14% more to develop than ENG software and 39% more than AIS software.

4.1.4 Cost Heuristics

Units for cost vary based on the office reporting them and the types of decisions that are being made. Engineering organizations often prefer to discuss things in technical units (e.g., requirement and SLOC) and

---

8 Burden labor rate is explained in Appendix G: Burden Labor Rate.

effort (e.g., hours or person months, months). Cost offices tend to communicate in terms of dollars and fiscal years. The following is a translation table that shows the same unit cost, production rate, and cost data expressed in different units.

Table 4: Unit Cost and Productivity

Project Size (40 KESLOC)	Unit Cost	Production Rate		
Domain	**Hours / KESLOC**	**Days / KESLOC**	**Hrs / Day**	**Cost / Day**
Real Time Software	1,007	25	40.4	$3,324
Engineering Software	936	26	35.4	$2,912
AIS Software	578	20	29.1	$2,393

Project Size (40 KESLOC)	Productivity				
Domain	**ESLOC / Hour**	**ESLOC /Day**	**People (FTEs)**	**Cost Month**	**Cost per Year**
Real Time Software	0.99	40	5.1	$99,720	$1,196,640
Engineering Software	1.07	38	4.4	$87,360	$1,048,320
AIS Software	1.73	50	3.6	$71,790	$861,480

Table 4 provides the unit cost (hours/KESLOC) and its inverse, productivity (ESLOC/hour). Depending on the type of information needed, one of the metrics may be preferred over the other. Alternatively, production rate is a metric that can be expressed in terms of units of product produced in a period of time (days/KESLOC) or units of time to produce a single product (ESLOC/day). It also provides monthly and annual costs by domain. The cost by year represents the annual costs for an average project for a full calendar year. This number doesn't help an engineering organization determine the total cost of a particular project, but it is a useful metric to technical managers when they are required to submit an annual budget.

4.2 Best-in-class/Worst-in-class

What differences are there between best-in-class and worst-in-class software projects?

This analysis examines the best- and worst-in-class projects within each of the three super-domains discussed in the previous section. To assess differences between projects, the three derived metrics explained in the previous section are used: unit cost, production rate, and cost.

4.2.1 Analysis Approach

An average size project within each super domain is used to derive unit cost, production rate, and cost. A ±1 standard error (SE) about the unit cost and production rate trend lines were used to identify best- and worst-in-class projects.

The definition of best-in-class and worst-in-class projects were developed as follows:

- Best-in-class projects: at or below the −1 SE value are projects that used less effort or less time to finish than an average project. This boundary represents the worst of the best-in-class projects—performance may actually be better.

- Worst-in-class projects: at or above the +1 SE value are projects that used more effort or more time to finish than an average project. This boundary represents the best of the worst-in-class projects—performance may actually be worse.

4.2.2 Real Time (RT) Software

4.2.2.1 Unit Cost

The average-size RT project (34,000 ESLOC for the RT domain) expends 39,664 labor hours of effort. Best-in-class projects expend 18,361 labor hours and worst-in-class projects expend 85,687 labor hours, a 10-fold increase. The difference between a best- or worst-in-class project from the average project is 21,304 labor hours. It is important to understand the context of the labor-hour differences in conjunction with project duration.

4.2.2.2 Production Rate

The average-size project delivers a product in 997 days (32.8 months). A best-in-class project delivers a product in 538 days (17.7 months). A worst-in-class project delivers a product in 1,848 days (60.8 months).

4.2.2.3 Cost

Table 5 summarizes the differences in unit cost and production rate between best-, average-, and worst-in-class RT projects. An average RT size project of 34,000 ESLOC was used to determine effort and schedule. Best-in-class RT projects are 2 times more efficient than average projects and 4.7 times more efficient than worst-in-class projects. Best-in-class projects are 1.8 times faster than an average projects and 3.4 times faster than a worst-in-class project. As mentioned earlier, the noted results for the best-in-class are the lowest reported numbers in the best-in-class bracket. Conversely, the reported results for worst-in-class are the highest reported numbers in the worst-in-class bracket.

Table 5: Real Time Software Best and Worst Summary

Metric	Best-in-class	Average	Worst-in-class
Effort (Labor Hours)	18,361	39,664	85,687
Schedule (Days)	538	997	1,848
Cost (per Day)	$2,805	$3,271	$3,813
Total Cost ($M)	$1.510	$3.262	$7.047

Using a burden labor rate of $150,000 per year,[9] the best-in-class project saves $1.752 million dollars over an average project and $5.537 million over a worst-in-class project.

[9] The burden labor rate is explained in Appendix G.

4.2.3 Engineering (ENG) Software

4.2.3.1 Unit Cost

There are 50 projects in the ENG super-domain. The average-size project (32,000 ESLOC for the ENG domain) expends 30,780 labor hours of effort. The best-in-class expends 14,468 labor hours and the worst-in-class expends 65,485, a 4.5 increase times the amount of best in class. The difference between a best- and worst-in-class project from the average project is 16,312 hours.

4.2.3.2 Production Rate

The best-in-class project delivers a software product in 640 days (21 months), an average project in 1,031 days (33.9 months), and a worst-in-class project in 1,659 days (54.6 months).

4.2.3.3 Cost

Table 6 summarizes the differences in unit cost and production rate between best, average, and worst-in-class ENG projects. An average ENG size project of 32,000 ESLOC was used to determine effort and schedule. The best-in-class ENG projects are 2.3 times more efficient than average projects and 5.3 times more efficient than worst-in-class projects. The best-in-class project is 1.6 times faster than an average project and 2.6 times faster than a worst-in-class project.

Table 6: Best and Worst Summary of Engineering Software

Metric	Best-in-class	Average	Worst in Class
Effort (Labor Hours)	14,468	30,780	65,485
Schedule (Days)	640	1,031	1659
Cost (per Day)	$1,859	$2,456	$3,246
Total Cost ($M)	$1.190	$2.531	$5.385

Best-in-class projects save $1.341 million dollars over average projects and $4.195 million dollars over a worst-in-class project.

4.2.4 Automated Information System (AIS)

4.2.4.1 Unit Cost

Using an average-size project of 72,000 ESLOC, best-in-class, average, and worst-in-class projects expended an average of 22,400, 39,114, and 68,297 labor hours of effort, respectively. There is a three-fold increase in effort between best and worst-in-class. The difference between a best or worst-in-class project and the average project is 16,713 labor hours.

4.2.4.2 Production Rate

The best-in-class average-size project delivers a product in 445 days (14.6 months). The average project delivers a product in 880 days (29 months). The worst-in-class a project delivers product in 1,743 days (57.3 months).

4.2.4.3 Cost

Table 7 summarizes the differences in unit cost and production rate between best, average, and worst-in-class projects. An average AIS size project of 72,000 ESLOC was used to determine effort and schedule. That makes best-in-class projects 1.7 times more efficient than average projects and 3 times more efficient than a worst-in-class projects. Best-in-class projects are 2 times faster than average projects and 4 times faster than worst-in-class projects.

Best-in-class projects save $1.375 million over average projects and $3.774M over worst-in-class projects.

Table 7: Best and Worst Summary of AIS Software

Metric	Best-in-class	Average	Worst-in-class
Effort (Labor Hours)	22,400 (% of avg)	39,114	68,297 (% of avg)
Schedule (Days)	445	880	1,743
Cost (per Day)	$4,144	$3,654	$3,223
Total Cost ($M)	$1.842	$3.217	$5.616

5 Project Planning, Trade-offs and Risk

In Sections 3 and 4, we showed how SRDR data could be used to provide a set of general characteristics for DoD projects and compared the three super domains based on those characteristics. In this section, we present our findings from a more extensive analysis of the data, where we explored correlations among requirements, size, duration, and effort. The goal of this work was to determine how well the data could be used to answer common questions related to planning or replanning software projects, such as

- How much growth should we plan for?
- How well can initial estimates be used to predict final outcomes?

The answers to the above questions are in the form of the following tables and graphs. Each is accompanied by a variety of statistics that are intended to help a reader make a reasonable assessment of the magnitude of growth, or in some cases reduction, in final actual values as compared to initial estimates. Also, they convey the uncertainty regarding such a prediction in the form a 95% prediction interval.

Each section first shows the equation describing the relationship between initial estimates and the final actual values obtained. The equations are then used to construct the following tables. The tables show columns for an initial estimate, the predicted final actual value, the percentage difference between the initial estimate and predicted final actual value, and the corresponding values of the upper and lower 95% prediction interval. The percentage difference changes with the initial estimate because of the nonlinear nature of the curves as shown in the figures.

Finally come the graphs which visually show the relationship between the initial estimates and the final actual values. For each of the graphs, the r2 for the plotted curve is reported. This value ranges from 0 to 1 and is interpreted as the percentage of the variation in dependent variable that is explained or accounted for by the variable in the independent variable. In the analysis, the initial estimate serves as the independent variable and the final actual value serves as the dependent variable. Values closer to 1 are better and indicate a highly systematic relationship. Values closer to 0 indicate a lack of relationship between the initial estimate and the corresponding final actual value. On the graphs, the forecast line is the value that would be predicted by any given input. Also shown are the upper and lower 95% prediction interval curves. These are useful for depicting the magnitude of uncertainty associated with making a prediction of the final actual value based on a given initial estimate. Finally, the graph also includes the dots representing the actual data used to fit the curves. They help to visualize the variation in the data which drives magnitude in the range between the upper and lower 95% prediction curves.

We present the strongest models we found to predict growth in requirements, ESLOC, schedule, and effort from the initial estimates. Each of the models can be used to construct predicted growth intervals for any given initial estimate, although we caution against using the model outside the bounds indicated by the 5th and 95th percentiles for each variable.

5.1 Estimation Relationships

Among the many factors and models for estimating effort, the SRDR data allows us to investigate the relationship between requirements and the size of the effort and then the relationship between the estimated size and the estimated effort as well as the final effort. A simple look at the correlations among requirements, size, duration, and effort found that the only actionable correlation was between size and effort.

Predicting Actual Total Effort by Estimated ESLOC

The following model shows that an initial estimate of ESLOC can also be used to predict the total actual effort. Although the model is only moderately strong, it is presented here in case an initial estimate of effort is not available, but an estimate of size (ESLOC) is available.

$\mathrm{n} = 163$

Regression Equation:

$$\ln Total\ Hours\_Actual = 2.031 + 0.8259 \ln ESLOC\_Estimated$$

which translates to:

$$Actual\ Total\ Hours = 7.614 * (Estimated\ ESLOC)^{.83}$$

The table shows the predictions have a "sweet spot" that is +/- 10% in the range from 75KESLOL to 200 KESLOC. The model accounts for over 67% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate of the initial by 158% at the low end (500 ESLOC) but an overestimate of -22% at the high end (500K ESLOC). This indicated that the model is reasonably good fit to the data.

Table 8: Prediction Values for Actual Total Hours (Effort) Using ESLOC

Initial ESLOC Estimate	Forecast Total Hours	Percent difference from Estimate	Prediction Interval – Total Hours	
			Lower 95%	Upper 95%
500	1,291	158%	264	6,305
750	1,805	141%	372	8,747
1,000	2,289	129%	475	11,040
2,500	4,879	95%	1,024	23,235
5,000	8,648	73%	1,828	40,911
7,500	12,088	61%	2,562	57,025
10,000	15,330	53%	3,255	72,213
25,000	32,675	31%	6,949	153,635
50,000	57,921	16%	12,300	272,755
75,000	80,961	8%	17,158	382,026
100,000	102,674	3%	21,717	485,437
150,000	143,515	-4%	30,249	680,898
200,000	182,006	-9%	38,248	866,094
300,000	254,403	-15%	53,200	1,216,562
400,000	322,634	-19%	67,199	1,549,009
500,000	387,926	-22%	80,526	1,868,786

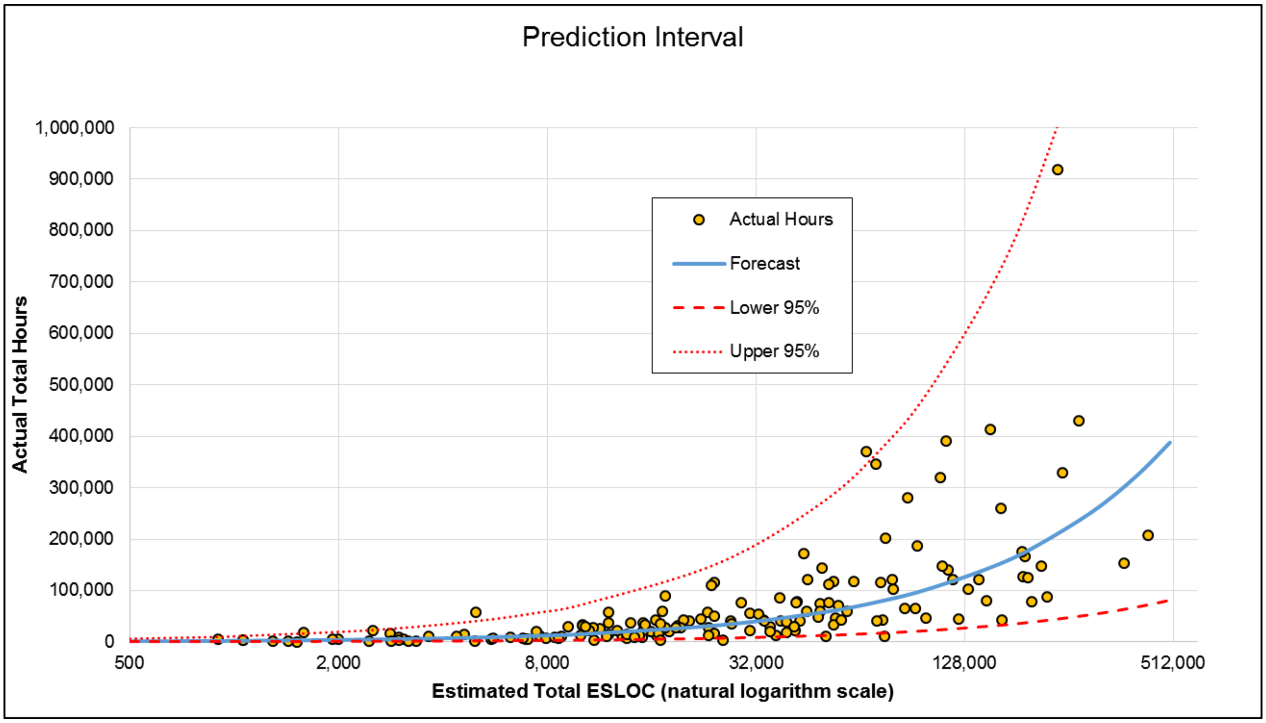


Figure 25: +/- 10 % is the range for 75,000 to 200,000 initial ESLOC estimates with +/- 10%

5.2 Software Growth – Predicting Outcomes

Can final outcomes be predicted from initial estimates?

This section describes the project performance as represented by 181 paired initial and final contractor submissions. As such, we measured the difference between the initial estimates and the actual outcomes. Section 5.2.1 describes the breakdown by Service and the age of the data. Section 5.2.2 explains our approach to modeling. Sections 5.2.3–5.2.7 present the statistical models for changes from estimates to actuals for total requirements, total software size (ESLOC), total duration (schedule), total effort hours (cost), and productivity (as measured by ESLOC[10] per person-month) for all records that had an initial SRDR paired with a final SRDR. Our intent is to present information to decision-makers regarding the usefulness of initial estimates in predicting project outcomes along these dimensions.

5.2.1 Description of Paired Initial/Final Submissions

Figure 26 shows the breakdown of paired submissions by service and their timelines. The initial reports were submitted between July 2001 and January 2013. The final reports were submitted between May 2003 and December 2012.

---

10 The definition and derivation of ESLOC (equivalent source lines of code) are explained in the Appendix B.

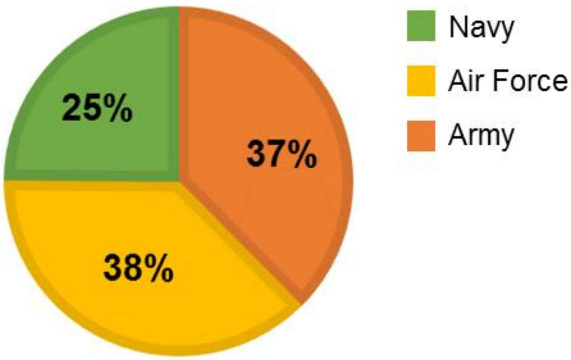


Figure 26: Paired Contractor Submissions by Service

The analysis dataset is spread across the three services (Marine Corps projects are included with Navy projects):

- Air Force (68)
- Army (68)
- Navy (45)

The submission dates for the paired data range from July 2001 to January 2013. There are a few projects from 2001 to 2004, but most of the projects are from the 2007 to 2012 timeframe.

Figure 27 shows the difference between the estimated end dates from the initial submissions to the actual end dates reported in the final submissions. As such, it represents the change in schedule.

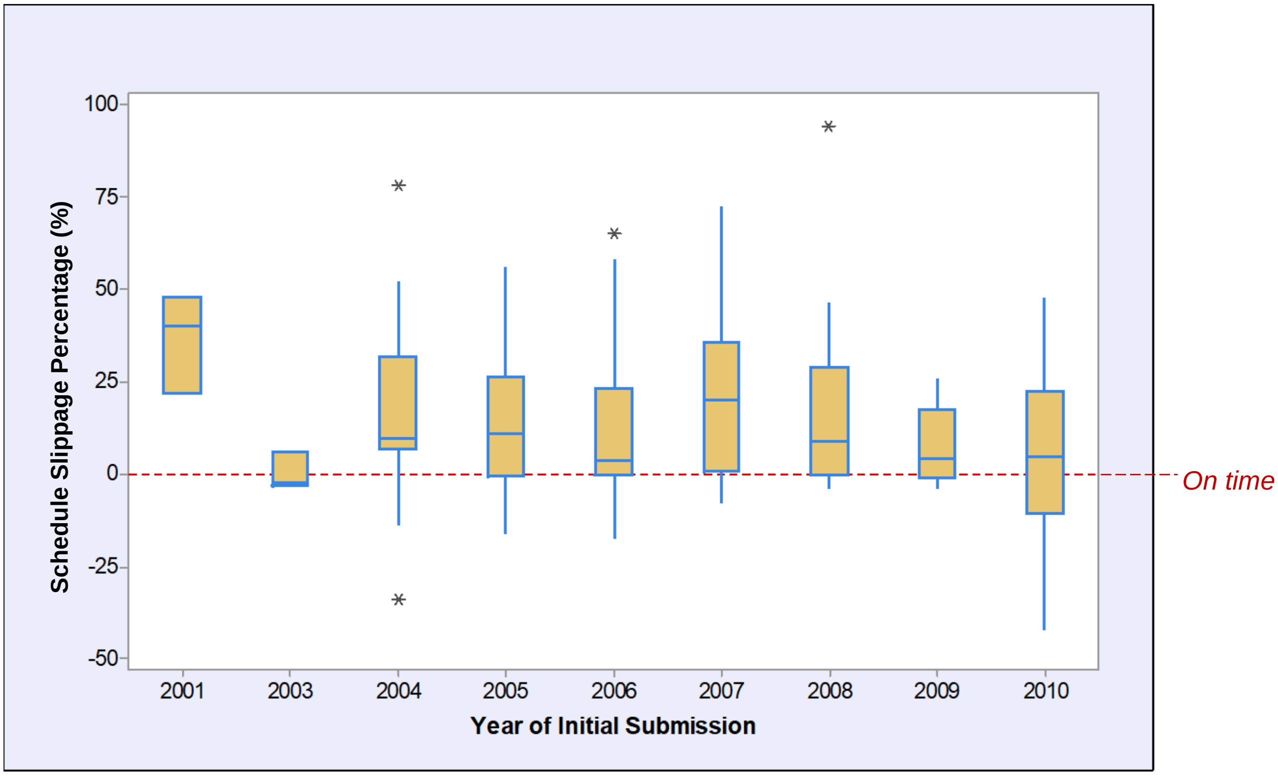


Figure 27: Schedule Slippage of Initial to Final Submission

Between 2004 and 2009, not a single Program finished early. Yet the center line for every year since 2003 is less than 20% overrun. This tends to suggest that Programs don't end early, occasionally they end on time, but most often they slip the planned end date from 0% to 20%.

5.2.2 Statistical Analyses

In Table 9 we show the value and percentage change using all cases for the five variables that form the focus of our analyses. The mean values are greater than the median values, indicating that the data is skewed. In this situation, the median (or 50th percentile) provides a better indication of the typical magnitude of change from the initial to final values as opposed to the mean (average) value. The median figures of percentage change provide a normalized indication of the magnitude of change. The variation between the initial and final values is evident by the wide ranges shown by the negative and positive percentage change columns, which represent over- and under-estimation in the initial submission.

Table 9: Change from Initial to Final Submission - All Cases

Comparison of Final Submission to Initial Submission[11] (Actual – Estimate)						
Change Variable	**Number of cases**		**Mean change**	**Median change**	**Largest negative % change**	**Largest positive % change**
Total Requirements	167	value	139	0	-100%	44,747%
		percent	469%	0%		

[11] Percentages are calculated as (Actual-Estimate)/Estimate.

Total ESLOC	181	value	24,816	6,399	-90%	1,440%
		percent	106%	42%		
Total Duration (Months)	181	value	15	9	-74%	625%
		percent	34%	8%		
Total Hours	180	value	16,487	4,651	-80%	1,162%
		percent	81%	19%		
Productivity (ESLOC/PM)	181	value	-32	-2	-96%	3,365%
		percent	34%	-1%		

Upon investigation, the 44747% increase appears to be due to the inconsistent use of a definition of a requirement. The initial definition equated a requirement to all changes made to a system. The final reported figure must have been based on a definition more closely resembling the number of changes made to the software. In other instances, it appears the scope of the project expanded significantly. These extremes are omitted by trimming the data to produce relationships that are more typical in the data. The analyses in the next section take this approach.

Table 10: Correlations of Change from Initial to Final Submission (Pearson Correlation Coefficients and *p-values*)

Change Category	**Total Requirements**	**Total ESLOC**	**Total Duration**	**Total Hours**
Total ESLOC	-0.067 *0.390*			
Total Duration	-0.027 *0.732*	0.112 *0.134*		
Total Hours	0.173 *0.025*	0.604 *0.000*	0.147 *0.049*	
Productivity	-0.075 *0.338*	0.251 *0.001*	0.138 *0.064*	-0.090 *0.228*

Table 10 shows very little correlation among these variables, which may seem counterintuitive. For example, given the enormous ranges of data for each of these variables, one might expect that when requirements increase during a project’s lifecycle that the ESLOC and schedule would also increase. The data, however, show that there are no discernible statistical patterns between these changes. Only the variability in Total Hours is moderately correlated with the variability in ESLOC, accounting for about 1/3 of the total variance.

The changes in Total Requirements, Total ESLOC, Total Duration (Months), Total Hours, and Productivity (ESLOC/PM) and their percentage changes were extensively investigated for relationships to other project attributes reported in the SRDR. Except where noted in the individual models presented later, statistical techniques (including analysis of variance, regression, correlation, and covariance analysis) failed to uncover any statistically significant relationships with the following attributes:

- project
- service (Army, Navy, Air Force)
- CMM/CMMI rating
- application domain
- super-domain
- development process
- personnel experience
- peak staff
- language
- requirements volatility
- negative and positive changes in productivity (using actual values minus estimated values)

Trimmed Data

After performing exploratory analyses on the full set of 181 paired cases, we found that extreme variability resulted in statistical models that yielded little predictive power. Each model evidenced extreme variability and resulted in many outliers. Rather than remove outliers—since we did not have access to substantive project information that might explain the circumstances behind any specific outlier—we instead chose to trim the extreme values for each of the five variables based on each variables' percentage change from initial estimate to final outcome.

We used the percentage change in each variable as the trim criteria so that cases which were less than the 5th percentile and greater than the 95th percentile were excluded for each variable in order to reduce the effects of extreme and possibly erroneous values. For example, the largest percentage growth in requirements was 44,747%, which seems highly suspicious. Each of the five variables (Total Requirements, Total ESLOC, Total Duration (Months), Total Hours, and Productivity (ESLOC/PM)) thus has its own dataset for each of the models presented in the following sections. The range of data values excluded are shown in the "percent change" histograms in the following sections.

Table 11 shows the descriptive statistics for the trimmed datasets used for statistical modeling in Section 2.3 - 2.7. This is a version of Table 1 based on trimming the lowest 5% and the highest 5% values. Much of the skewness was trimmed, but further analysis yielded predictive models of low or moderate usefulness. This led us to investigating transformations of the original data. As discussed below, nonlinear models provided a strong ability to predict the final outcomes.

Table 11: Change from Initial to Final Submission – Trimmed Cases

Change of Final Submission from Initial Submission (Trimmed datasets) (Actual – Estimate)						
Change Variable	number of cases		Mean change	Median change	Minimum change (5^{th} percentile)	Maximum change (95^{th} percentile)
Total Requirements	150	value	-104	0	-5,635	6047
		percent	1%	0%	-76%	176%
Total ESLOC	162	value	22,752	6,686	-164,672	603,536
		percent	64%	42%	-61%	420%

Total Duration (Months)	161	value	15	9	-17	78
		percent	20%	8%	-37%	155%
Total Hours	162	value	15,256	4,505	-56,778	339,697
		percent	50%	19%	-45%	453%
Productivity (ESLOC/PM)	162	value	-18	-2	-1,094	269
		percent	7%	-1%	-75%	150%

Sections 2.3 to 2.7 present the results of statistical modeling for predictive purposes using the initial estimates to predict the final outcomes. We found that the models of greatest utility were non-linear models based on natural logarithm transformations of both the initial and final values, of the form

$$\boldsymbol{Y = cX^{\beta}\epsilon}$$

which translates to the regression model

$$\ln y = \ln c + \beta \ln x + \ln \epsilon$$

Where y = the actual (final) outcome, c = constant, x = the initial estimate, β = the regression coefficient of the natural logarithm model, and $\boldsymbol{\epsilon}$ represents the error term. For this particular type of model (both x and y transformed to natural log values), the coefficient β represents an elasticity (in economic terms); that is, a 1% change in x roughly equals a β% change in y. When translated from the natural log model, β is the exponent to the initial estimate X.

Of course, estimators and decision makers want to more accurately predict the growth of project software which is often a cost and schedule driver. As shown in the following sections, using the initial estimate values we can predict the outcome for Total Requirements, Total ESLOC, total schedule duration (months), total effort hours, and productivity using the initial estimates. We present the fit and equation for each model and include a table of the forecast values with their associated predication intervals based on a range of input values. This allows the reader to roughly gauge the expected outcomes based on an initial estimate.[12] We also illustrate the model with a plot of the derived prediction intervals. The full statistical results for each model along with a scatterplot of the model's fit can be found in Appendix F.

All the models presented in Sections 2.3 to 2.7 use only one independent variable (x) for one dependent variable (y). As mentioned earlier, we found that adding more variables did not improve the models and usually degraded the fit. This means that the r2 statistic also represents the squared Pearson correlation between the x and y variables, so that when r^2 equals .9, .8, or .7, the corresponding correlation coefficient equals .95, .89, or .84, respectively.

Each of the models presented here show the number of cases, the original natural logarithm Minitab equation, the translated equation, and the r^2 statistic. We also include a table of nominal values for the input estimate (x), the predicted (forecast) value (y'), the percentage difference between the predicted value and the estimate, and

12 Statistical software enables the direct calculation of the forecast value and prediction interval for a given input value. Prediction intervals are the appropriate statistic to use for the forecast of a new data point. We used a 95% confidence level for the prediction intervals. We also present a prediction interval table for effort hours based on a 70% confidence level to show the trade-off in accuracy when certainty decreases.

the prediction interval surrounding the predicted value. The table is followed by a scatterplot of the actual data (yi) values against the predicted regression line plus the prediction interval.

The tables can be used to get a quick rough estimate of a final outcome for new cases by interpolating for a new value. Although this will yield a ball park prediction, the tables are not fine-grained enough to account for the non-linearity. For this reason we recommend that the actual equation be used. For even greater confidence in estimating a new case, please contact us for a copy of our datasets which then can be used with statistical software to reproduce the models and outputs. We are allowed to share our data, with the DoD cost community and do so using the U.S. Army AMRDEC SAFE website for secure transfer of files.

5.2.3 Total Requirements

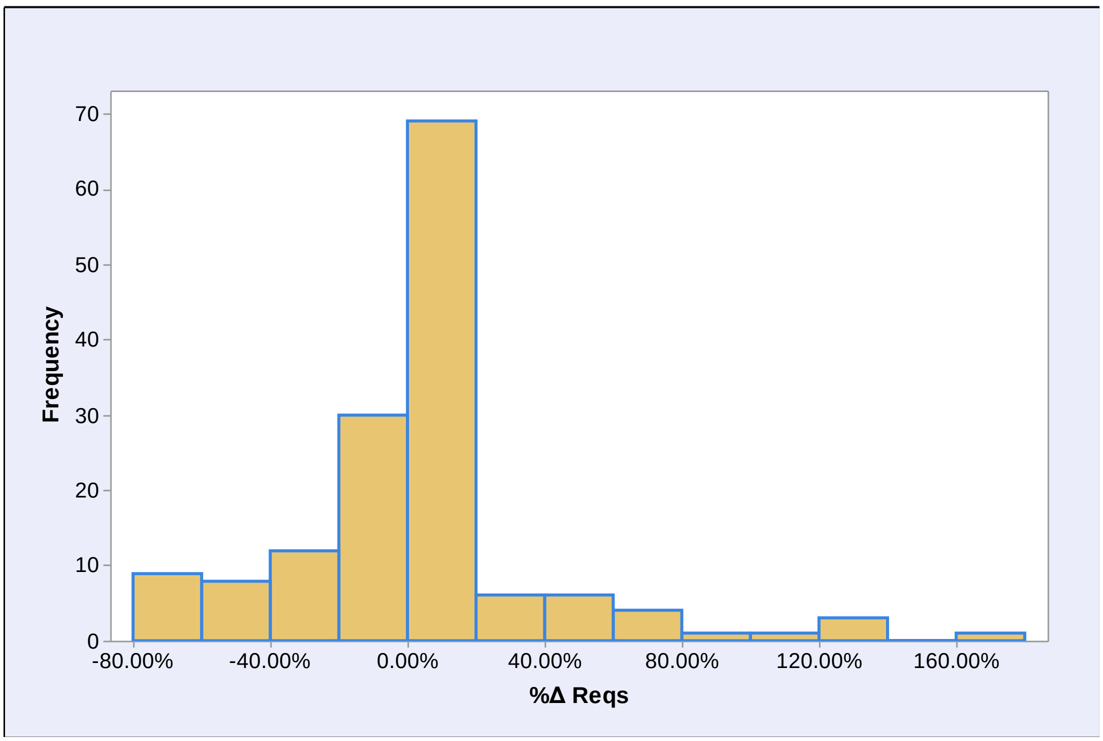


Figure 28: Percentage Difference in Actual versus Estimated Total Requirements

The percentage difference in estimated total requirements versus the actual total requirements (Table 3) shows the median percentage change in requirements to be zero. However the minimum and maximum values show that changes can range from -5,635 to 6,047 total requirements. Of the 150 cases, 59 cases showed a decrease in total requirements from the original estimate, 55 showed an increase, and 36 showed no change. All three services (Army, Navy, and Air Force) showed a median percentage change of 0%. Projects with negative or positive change in productivity also showed median percentage changes of zero. When considering the three super-domains, AIS, ENG, RT, the median percentage change for each was zero. Consideration of service, change in productivity, or super domain does not provide any additional information.

For predictive purposes, the following model provides a very strong fit in predicting the total actual (final) requirements given only the initial estimate. The results of the regression model on the transformed data is presented below:

n = 148

Regression Equation:

$$\textit{ln Total Reqts\_Actual} = 0.250 + 0.9456 \textit{ ln Total Reqts\_Estimated}$$

which translates to:

$$Actual\ Total\ Reqts = 1.28 * (Estimated\ Total\ Reqts)^{.95}$$

The constant, 1.28, indicates that for small projects there is roughly a 28% increase in requirements from initial estimates to final values. However, as the number of requirements increases, the percentage increase is reduced by the exponent, 0.95 when applied to the number of initial requirements. The first two columns in Table 12 show requirements growth becoming inverted at 100.

The adjusted r^2 equals .936; the model accounts for over 93% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Based on this model we see that requirements are underestimated for very low numbers and overestimated for most of the range of data, Here, predicted values show an underestimate by the initial submission of 16% at the low end (6 requirements) but show an overestimate of 23% at the high end (12,000 requirements), with the inflection point at 100 requirements.

Table 12: Prediction Interval Values for Total Requirements

Initial Requirements Estimate	**Forecast Requirements**	**Percent difference from Estimate**	**Prediction Interval**	
			Lower 95%	Upper 95%
6	7	16%	3	15
10	11	13%	5	25
25	27	8%	12	59
50	52	4%	24	113
75	76	2%	35	166
100	100	0%	46	218
250	238	-5%	109	517
500	458	-8%	210	996
750	672	-10%	309	1,462
1,000	882	-12%	405	1,921
2,000	1,699	-15%	778	3,707
5,000	4,040	-19%	1,843	8,857
10,000	7,781	-22%	3,534	17,134
12,000	9,245	-23%	4,193	20,385

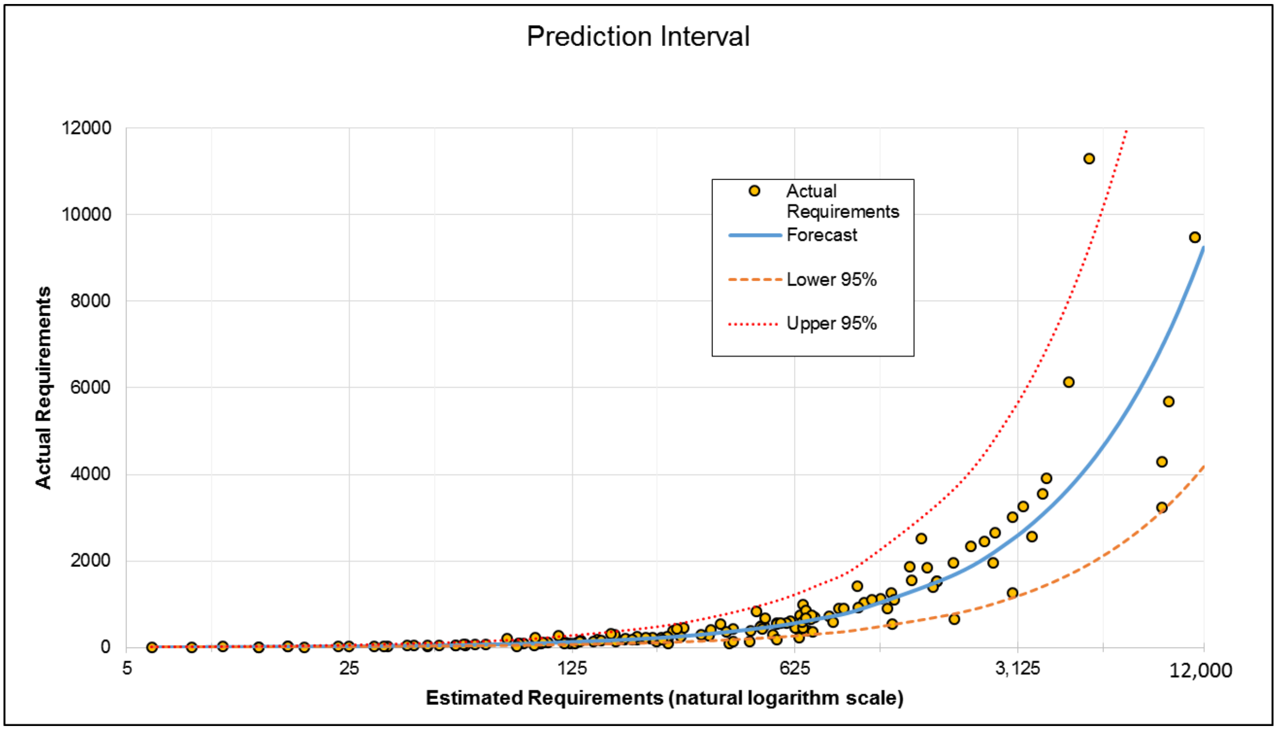


Figure 29: Prediction Interval for Actual Total Requirements

The data suggests that planned total requirements tends to hold true and is a fairly good predictor of the total number of requirements when the project is complete. It also indicates a slight tendency to under estimate requirements when the planned number of requirements are few (i.e., less than 100) and a slight tendency to overestimate the total number of requirements when the planned number of requirements is over 100. For practical purposes, projects should plan a software project (i.e., build, increment, or release) to consist of 80-120 requirements, adding additional projects, as needed to accommodate more requirements.

5.2.4 Total ESLOC

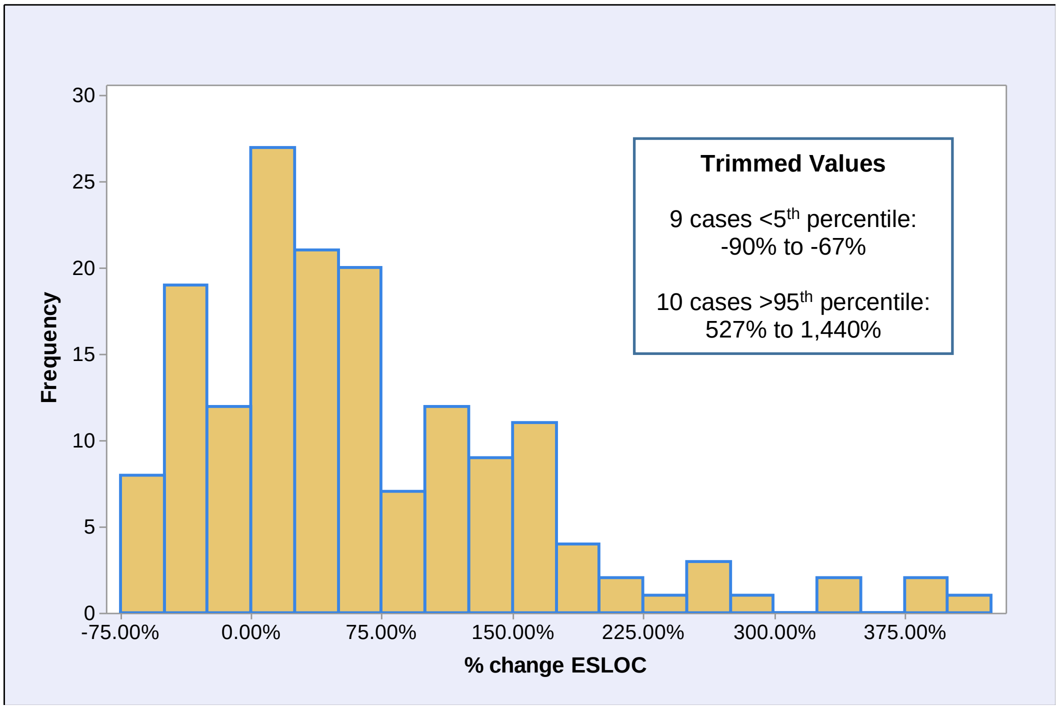


Figure 30: Percentage Difference in Actual versus Estimated Total ESLOC

Referring to Table 13, the change in total ESLOC shows a 42% median percentage increase in software size. The mean percentage change was 64%, indicating the data is skewed toward zero. The minimum amount of change (actual minus estimated) was -164.672 ESLOC and the maximum was 603,536. There were 39 cases that showed a decrease from the estimated size, 121 that showed an increase, and 2 that showed no change. Projects with a negative change in productivity showed a median increase of 7%, but projects with a positive change showed a 79% increase. The Army, Navy, and Air Force all had projects with median size increases of 48%, 43%, and 38%, respectively.

Projects segmented into the three super-domains all showed positive median size increases. AIS increased by 70%, RT increased by 38%, and ENG increased by 28%. For predictive purposes, the following model provides a very strong fit in predicting the total actual (final) ESLOC given only the initial estimate. The results of the regression model on the transformed data is presented below:

n = 162

Regression Equation:

$$ln\ ESLOC\_Actual = 0.701 + 0.9640\ ln\ ESLOC\_Estimated$$

which translates to:

$$Final\ Total\ ESLOC = 2.02 * (Initial\ ESLOC)^{.96}$$

The adjusted r^2 equals .849; the model accounts for over 84% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. The model shows that ESLOC is underestimated for the entire data range. Predicted values show an underestimate by the initial submission of 71% at the low end (100 ESLOC) decreasing to a 26% underestimate at the high end (500,000 ESLOC).

Table 13: Predicted Values for Total ESLOC

Initial ESLOC Estimate	Forecast ESLOC	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
100	171	71%	53	551
500	806	61%	256	2,532
1,000	1,572	57%	504	4,898
2,000	3,066	53%	990	9,490
5,000	7,416	48%	2,411	22,806
10,000	14,466	45%	4,718	44,357
15,000	21,385	43%	6,981	65,510
25,000	34,991	40%	11,426	107,156
50,000	68,257	37%	22,266	209,242
100,000	133,148	33%	43,316	409,280
250,000	322,064	29%	104,127	996,147
500,000	628,247	26%	201,776	1,956,102

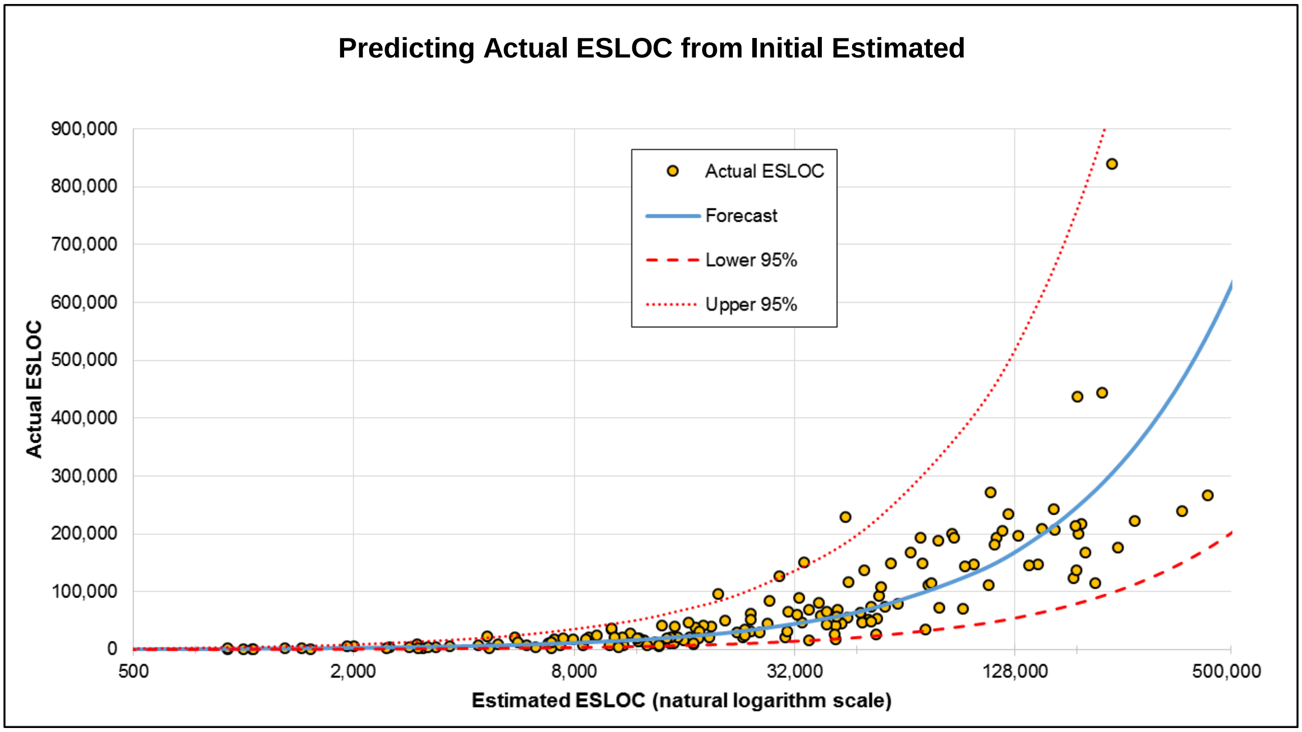


Figure 31: Prediction Interval for Actual Total ESLOC

In practice, a 30% size growth factor has been widely used as a rule of thumb. Without a reference data to back up the rule of thumb, it has been dismissed during contract awards and negotiations. This data corroborates that rule of thumb for projects around 250 KESLOC. It also suggests it is overly conservative for smaller projects. Based on this data set, 25% size growth at a minimum, should be integrated into a project's software estimation process.

5.2.5 Total Duration (Schedule)

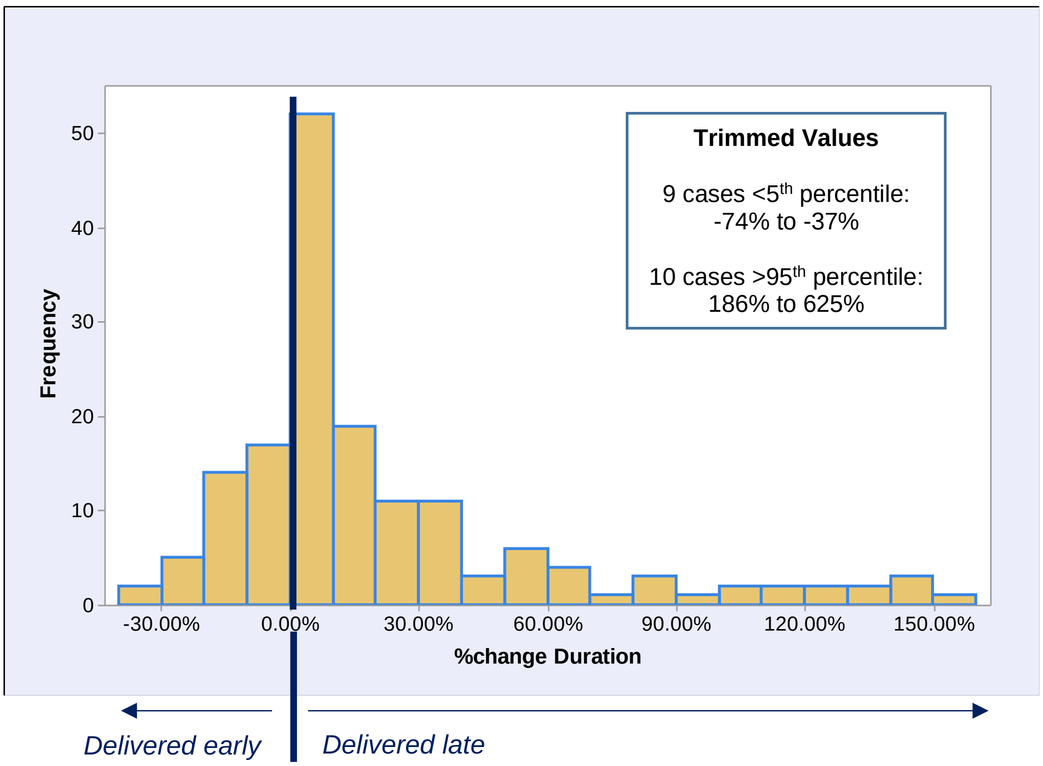


Figure 32: Percentage Difference in Actual versus Estimated Total Duration (Months)

Duration is measured as the start of requirements until the last phase is conducted as reported on the SRDR Form 2630-3. Referring to Table 14, total duration percentage change shows an overall positive median increase of 8%. The mean change percentage is 20% indicating the data is skewed toward zero. The change in months of duration ranged from -17 to 78. There were 38 cases that showed a decrease from the estimate, 88 that showed an increase, and 35 that showed no change. Projects with a positive change in productivity showed a median value increase of 10% in duration while projects with negative productivity had a median change of zero.

The grouping of the data by super-domain does not provide any additional information. The AIS, ENG, and RT, super-domains have a 0%, 2%, and 11% change in duration, respectively. Each super-domain's minimum and maximum values overlap with the other super-domains.

The Army, Navy, and Air Force services showed 1%, 18%, and 0% change in schedule duration.

For predictive purposes, the following model provides a moderately strong fit in predicting the total actual (final) schedule duration given only the initial estimate. However, the addition of a services variable (Army, Navy, and Air Force) also proved statistically significant but did not add to the overall fit of the model. Instead we subdivided the data into three datasets – one for each service. The results show a different model for each of the services and are presented below with their corresponding prediction tables and graphs, following the result for the overall model.

ALL Cases

n = 161

Regression Equation:

$$\mathit{ln\ Months\_Actual} = 0.8352 + 0.7878\ \mathit{ln\ Months\_Estimated}$$

which translates to:

$$Actual\ Total\ Duration = 2.31 * (Estimated\ Total\ Duration)^{.79}$$

The adjusted r^2 equals .776; the model accounts for over 77% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. As with the requirements model, the duration model shows underestimated values at the low end and overestimated values at the high end. Here, predicted values show an underestimate by the initial submission of 131% at the low end (5 months) but show a -17% overestimate at the high end (120 months), with the inflection point at about 50 months.

Table 14: Predicted Values for Schedule Duration – All cases

Estimated Total Months	Forecast Total Months	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
5	8	**64%**	5	14
8	12	**48%**	7	20
12	16	**36%**	10	27
15	19	**30%**	12	32
20	24	**22%**	15	40
25	29	**16%**	18	48
30	34	**12%**	20	56
35	38	**8%**	23	63
40	42	**5%**	25	70
45	46	**3%**	28	77
50	50	**1%**	30	83
60	58	**-3%**	35	96
70	66	**-6%**	39	109
80	73	**-9%**	44	121
90	80	**-11%**	48	133
100	87	**-13%**	52	145
110	94	**-15%**	56	156

120	100	**-17%**	60	167

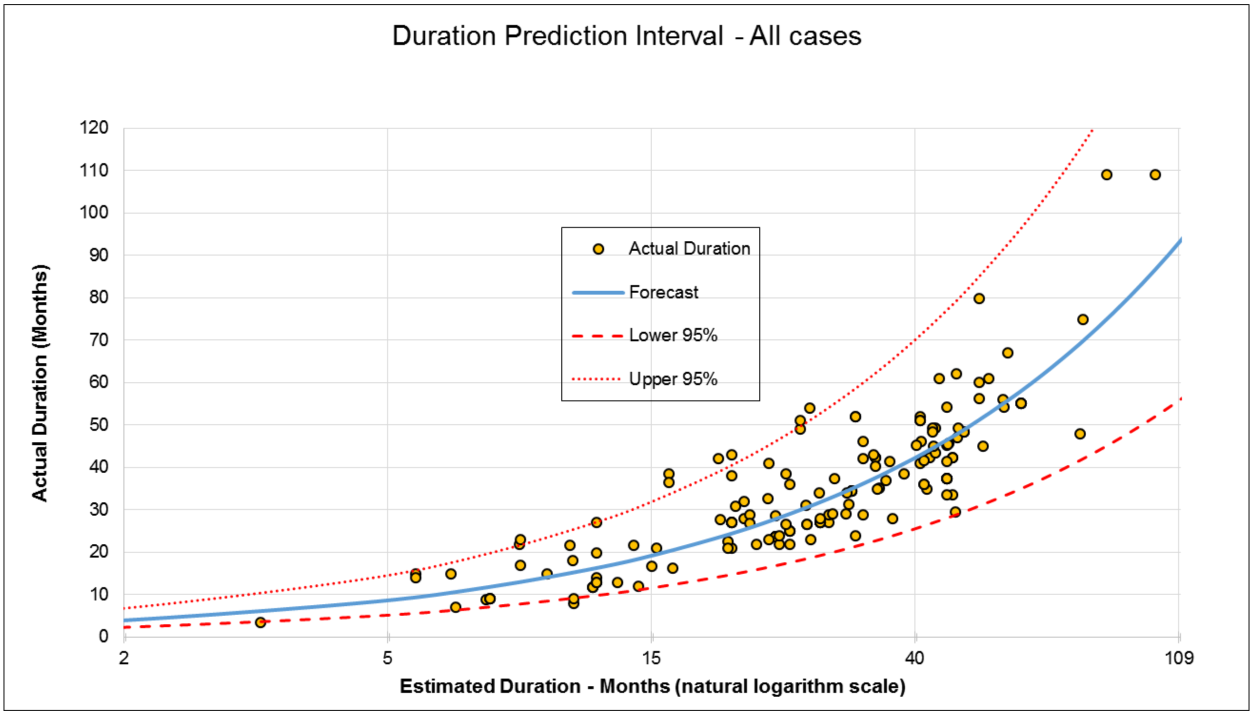


Figure 33: Prediction Interval for Actual Total Duration (Months)

The data suggests that projects planned for less than 3 years (i.e., 36 months) tend to finish 3-4 months late. And projects planned for more than 3 years tend to finish early, on time, or marginally late (i.e., less than 1 month). Without further research into why this tends to be the case, it is unknown what drivers this outcome. It is most likely a combination of engineering, management, and funding factors. Although a project may resist planning a schedule slip, this data does provide a basis for quantifying the impact associated with the risk of a slippage. It does seem to imply that given more time, a project has the opportunity to react and revise their plan, the greater the probability of finishing the project within the planned duration.

> ***Army – Schedule Duration***
>
> n = 65
>
> Regression Equation:
>
> $$\ln Months\_Actual = 0.5146 + 0.8657 \ln Months\_Estimated$$
>
> which translates to:
>
> ***Army:*** $Actual\ Total\ Duration = 1.67 * (Estimated\ Total\ Duration)^{.87}$

The adjusted r^2 equals .829; the model accounts for over 82% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate by the initial submission of 35% at the low end (5 months) but show a -12% overestimate at the high end (120 months), with the inflection point at 45 months.

Table 15: Predicted Values for Schedule Duration - Army

Estimated Total Months	Forecast Total Months	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
5	7	35%	4	11
8	10	27%	6	16
12	14	20%	9	23
15	17	16%	11	28
20	22	12%	14	35
25	27	9%	17	43
30	32	6%	20	50
35	36	4%	23	57
40	41	2%	26	65
45	45	0%	28	72
50	49	-1%	31	78
60	58	-3%	36	92
70	66	-5%	41	106
80	74	-7%	46	119
90	82	-9%	51	132
100	90	-10%	56	145
110	98	-11%	61	158
120	106	-12%	65	171

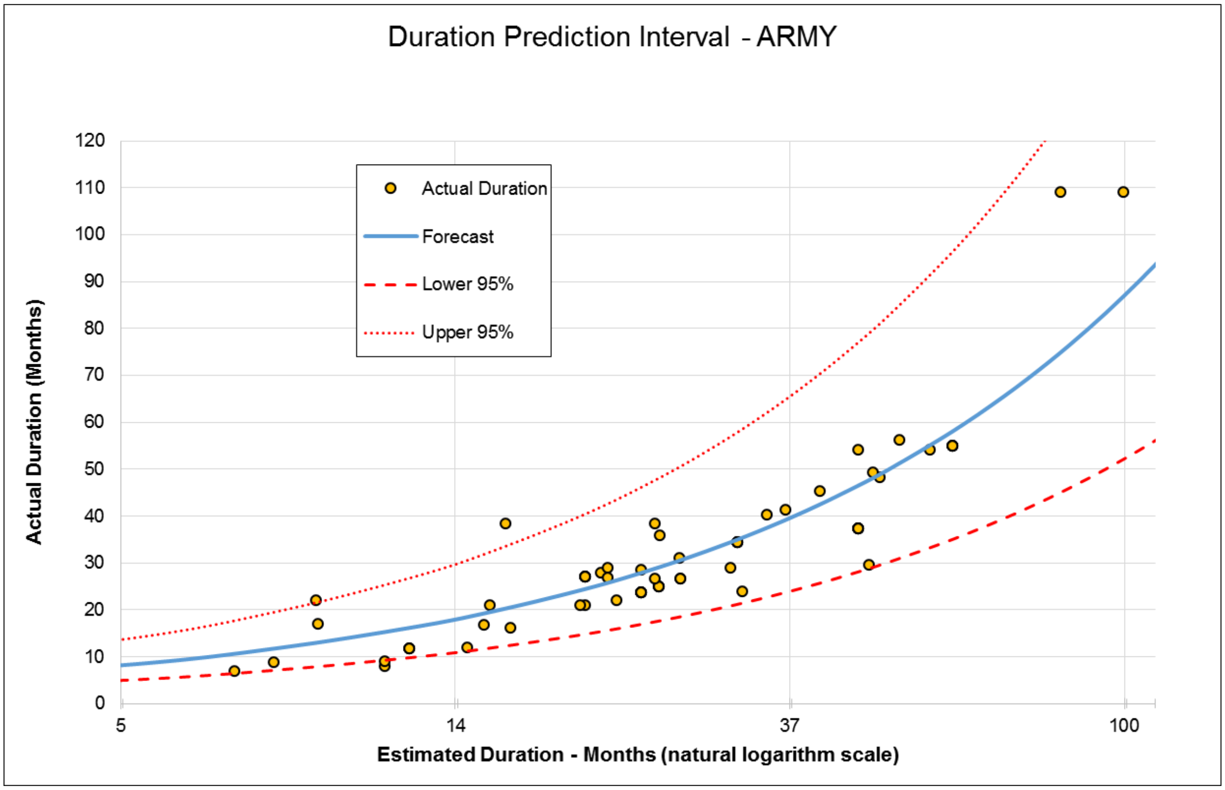


Figure 34: Prediction Interval for Actual Total Duration (Months) - Army

The Army data suggests that projects planned for less than 3 years (i.e., 36 months) tend to finish 2 months late. Which is less of a slip compared to the 3-4 month slip when considering all the data. And projects planned for more than 3 years tend to finish on time or early.

Air Force – Schedule Duration

n = 39

Regression Equation:

$$\ln Months\_Actual = 1.085 + 0.7258 \ln Months\_Estimated$$

which translates to:

Air Force: $Actual\ Total\ Duration = 2.96 * (Estimated\ Total\ Duration)^{.73}$

The adjusted r^2 equals .601; the model accounts for over 60% of the variance. Below are the predicted (forecast) values for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate by the initial submission of 90% at the low end (5 months) but an overestimate of -20% at the high end (120 months), with the inflection point at about 49 months.

Table 16: Predicted Values for Schedule Duration - Air Force

Estimated Total Months	Forecast Total Months	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
5	10	90%	5	18
8	13	67%	8	24
12	18	50%	10	31
15	21	41%	12	36
20	26	30%	15	44
25	31	22%	18	51
30	35	16%	21	59
35	39	12%	23	65
40	43	8%	26	72
45	47	4%	28	79
50	51	1%	30	85
60	58	-4%	34	98
70	65	-8%	38	111
80	71	-11%	41	123
90	78	-14%	44	135
100	84	-16%	48	147
110	90	-18%	51	159
120	96	-20%	54	170

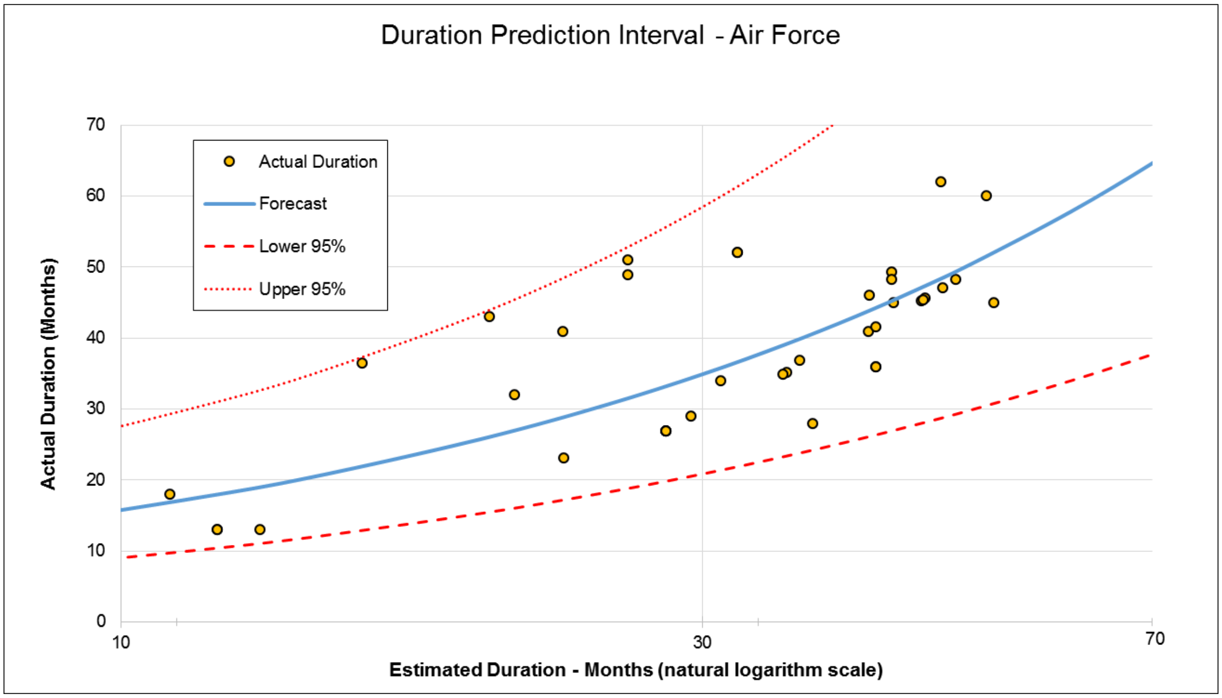


Figure 35: Prediction Interval for Actual Total Duration (Months) - Air Force

The Air Force data suggests that projects planned for less than 3 years (i.e., 36 months) tend to finish 4-6 months late, whereas Army projects planned for less than 3 years (i.e., 36 months) tend to finish 3-4 months late. Generally, projects planned for more than 4 years tend to finish early.

Navy – Schedule Duration

n = 57

Regression Equation:

$$\ln Months\_Actual = 1.036 + 0.7410 \ln Months\_Estimated$$

which translates to:

Navy: $$Actual\ Total\ Duration = 2.8182 * (Estimated\ Total\ Duration)^{.7410}$$

The adjusted r^2 equals .793; the model accounts for over 79% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate by the initial of 86% at the low end (5 months) but show a -18% overestimate at the high end (120 months), with the inflection point at about 55 months.

Table 17: Predicted Values for Schedule Duration - Navy

Estimated Total Months	Forecast Total Months	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
5	9	86%	5	16

8	13	64%	8	23
12	18	48%	10	31
15	21	40%	12	36
20	26	30%	15	45
25	31	22%	18	53
30	35	17%	20	60
35	39	12%	23	68
40	43	8%	25	75
45	47	5%	27	82
50	51	2%	30	88
60	59	-2%	34	101
70	66	-6%	38	114
80	72	-9%	42	126
90	79	-12%	45	138
100	86	-14%	49	150
110	92	-17%	52	161
120	98	-18%	56	172

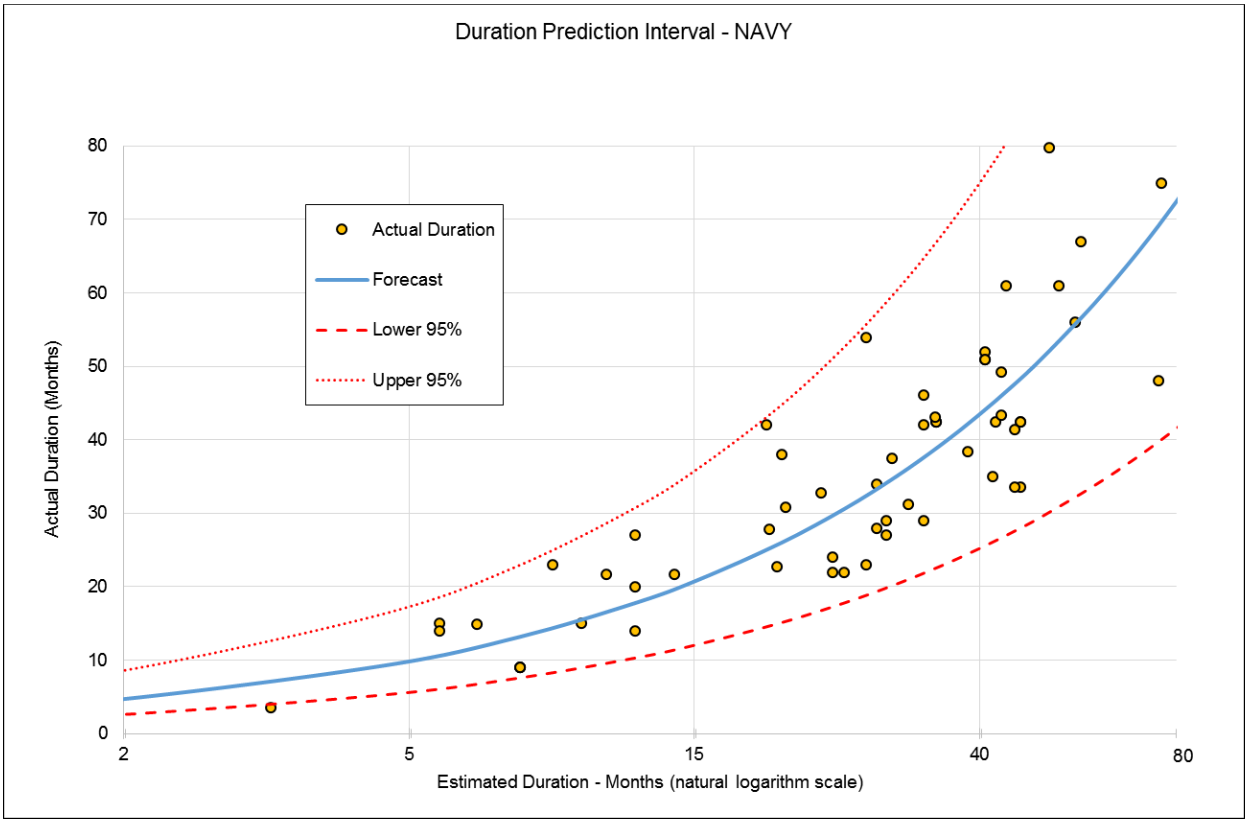


Figure 36: Prediction Interval for Actual Total Duration (Months) - Navy

The Navy data produced a better variance, yet comparing the estimate to the forecasted number in Table 17 shows the Navy data, like the Air Force, indicate that projects planned for less than 3 years (i.e., 36 months) tend to finish 4-6 months late, which is a longer lag time than the what was observed in the Army data. For projects planned for more than 5 years, the data suggest that projects tend to finish on time or early.

5.2.6 Total Hours (Effort)

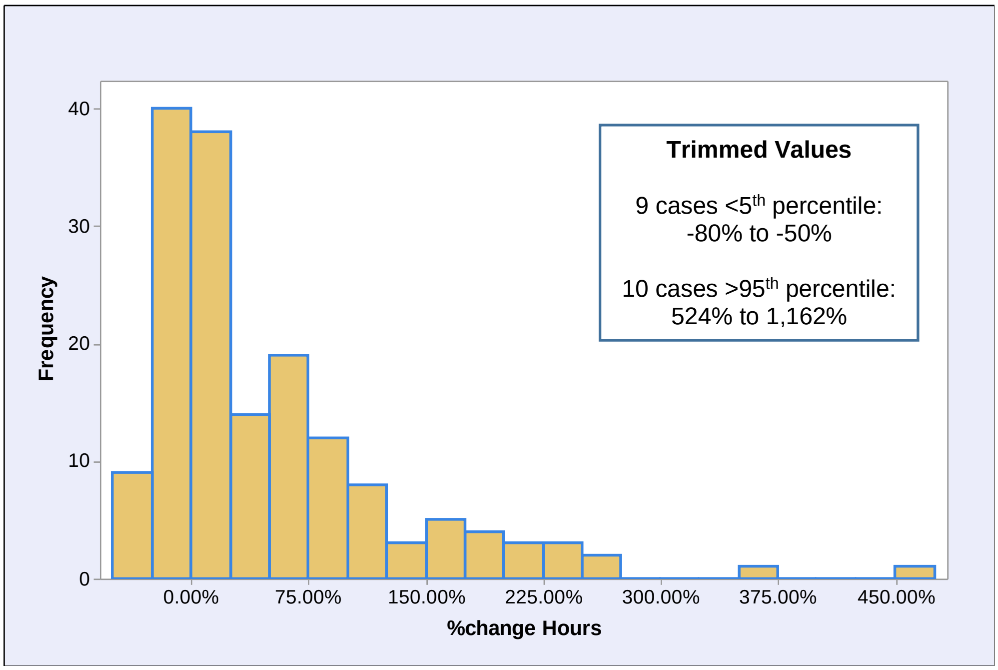


Figure 37: Percentage Difference in Actual versus Estimated Total Hours

Referring to Table 18, the median increase in hours between initial and final SRDRs was 19% overall (based on actual minus estimated values). The overall mean was 61%. The minimum value for the change in hours was -97,652 and the maximum increase was 350,591. There were 49 cases that showed a decrease from the initial estimate, 111 that showed an increase, and 2 that showed no change. Negative and positive productivity groups showed a 51% and 6% median increase in hours respectively. It makes sense that negative productivity groups expend more hours than positive productivity groups.

The grouping of data by service showed about a 25% median increase in hours for the Army, a 19% increase for the Navy, and a 16% increase for the Air Force. Grouping by super-domains showed an 18% median increase in hours for AIS, a 37% increase for ENG, and a 22% increase RT. Neither of these factors offered any statistical insight into the differences in increased hours between initial and final SRDRS.[13]

The results of the regression model on the transformed data is presented below:

[13] Although the difference in means for the super domains were also suggestive, analysis of variance (ANOVA) proved negative.

n = 162

Regression Equation:

$$ln\ Total\ Hours\_Actual = 1.198 + 0.9097\ ln\ Total\ Hours\_Estimated$$

which translates to:

$$Actual\ Total\ Hours = 3.31 * (Estimated\ Total\ Hours)^{.91}$$

The adjusted r^2 equals .898; the model accounts for over 89% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate by 89% at the low end (500 hours) but an overestimate of -5% at the high end (1 million hours), with the inflection point at about 577,500 total hours.

Table 18: Prediction Values for Actual Total Hours (Effort) - 95% Confidence level

Initial Total Hours Estimate	**Forecast Total Hours**	**Percent difference from Estimate**	**Prediction Interval**	
			Lower 95%	**Upper 95%**
500	945	89%	391	2,285
1,000	1,776	78%	739	4,267
5,000	7,677	54%	3,226	18,268
10,000	14,423	44%	6,074	34,246
50,000	62,361	25%	26,267	148,055
100,000	117,157	17%	49,246	278,719
250,000	269,637	8%	112,815	644,452
500,000	506,559	1%	210,898	1,216,711
750,000	732,526	-2%	303,925	1,765,551
1,000,000	951,660	-5%	393,779	2,299,912

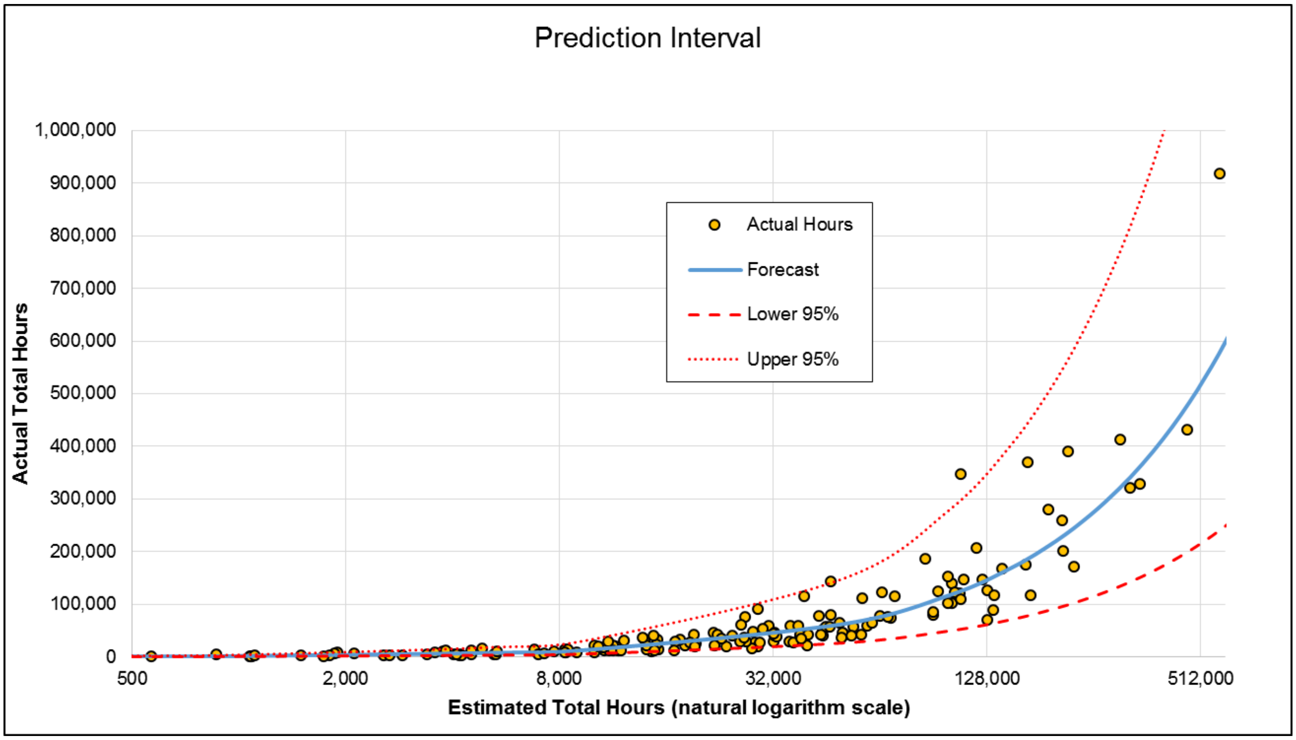


Figure 38: 95% Prediction Interval for Actual Total Hours (Effort)

The following table and graph are presented to contrast the forecast and the prediction intervals of actual total hours using a 70% confidence level for the prediction rather than a 95% confidence level. Forecast values remain the same; only the interval for predicting new cases changes. Note how the intervals are narrowed when we reduce the surety. The graph also reflects the increased risk of inaccuracy by showing that many of the cases now fall outside of the predicted intervals. Decision makers should be aware of this trade-off when judging the range of outcomes for any variable.

Table 19: Prediction Values for Actual Total Hours (Effort) - 70% Confidence level

Initial Total Hours Estimate	**Forecast Total Hours**	**Percent difference from Estimate**	**Prediction Interval**	
			Lower 70%	**Upper 70%**
500	945	89%	594	1,504
1,000	1,776	78%	1,119	2,817
5,000	7,677	54%	4,864	12,118
10,000	14,423	44%	9,148	22,740
50,000	62,361	25%	39,555	98,316
100,000	117,157	17%	74,232	184,903
250,000	269,637	8%	170,428	426,595
500,000	506,559	1%	319,348	803,521
750,000	732,526	-2%	460,964	1,164,072
1,000,000	951,660	-5%	598,010	1,514,452

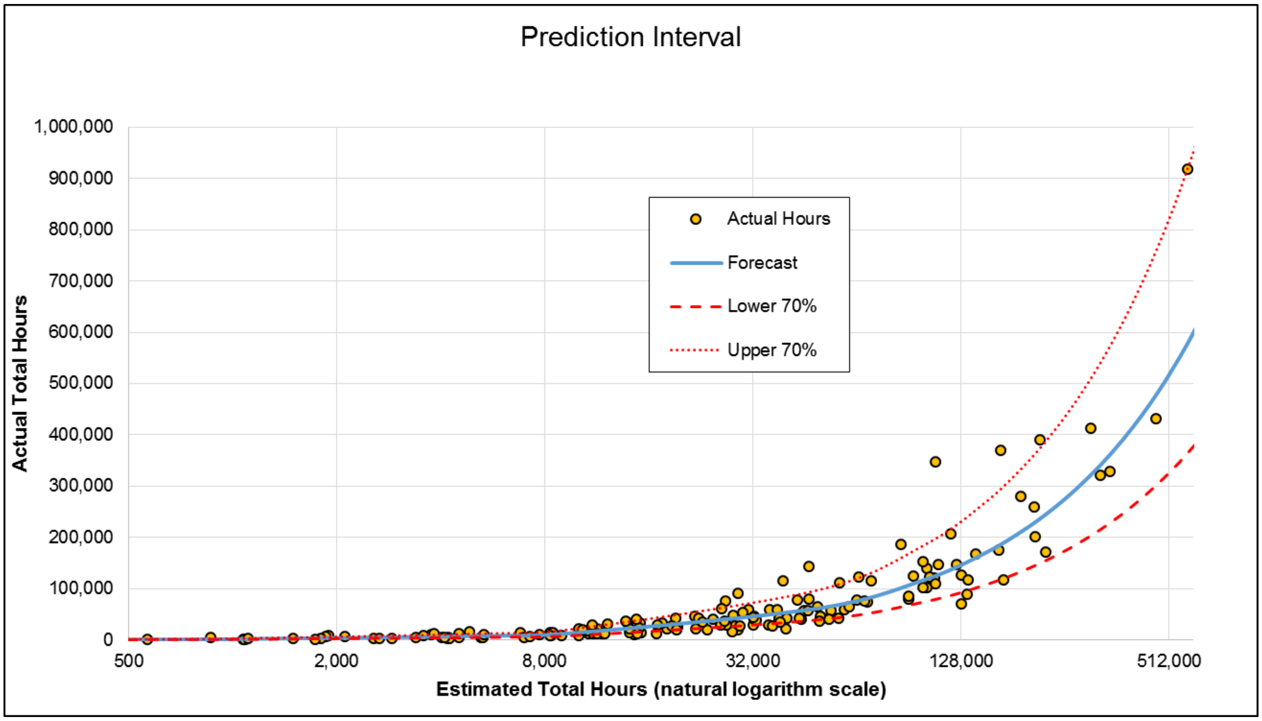


Figure 39: 70% Prediction Interval for Actual Total Hours (Effort)

Using a 95% confidence bound produces a range that is useless in practice. As shown above, if the initial effort is estimated to be 100,000 hours then based on the SRDR data set, the forecasted actual hours is 117,157. And based on the Upper 95% Confidence interval the project is unlikely to exceed 278,719 hours. For planning purposes this number is essentially useless. A program manager cannot plan to hold over two times the point estimate budget in risk reserve. A more practical approach is to reduce the confidence interval to a reasonable level. As shown in Table 19, the confidence interval was lowered to 70%. This yields an upper prediction interval of 184,903 hours. This value is less than the planned budget and may be a more useful number for quantifying risk.

5.2.7 Productivity

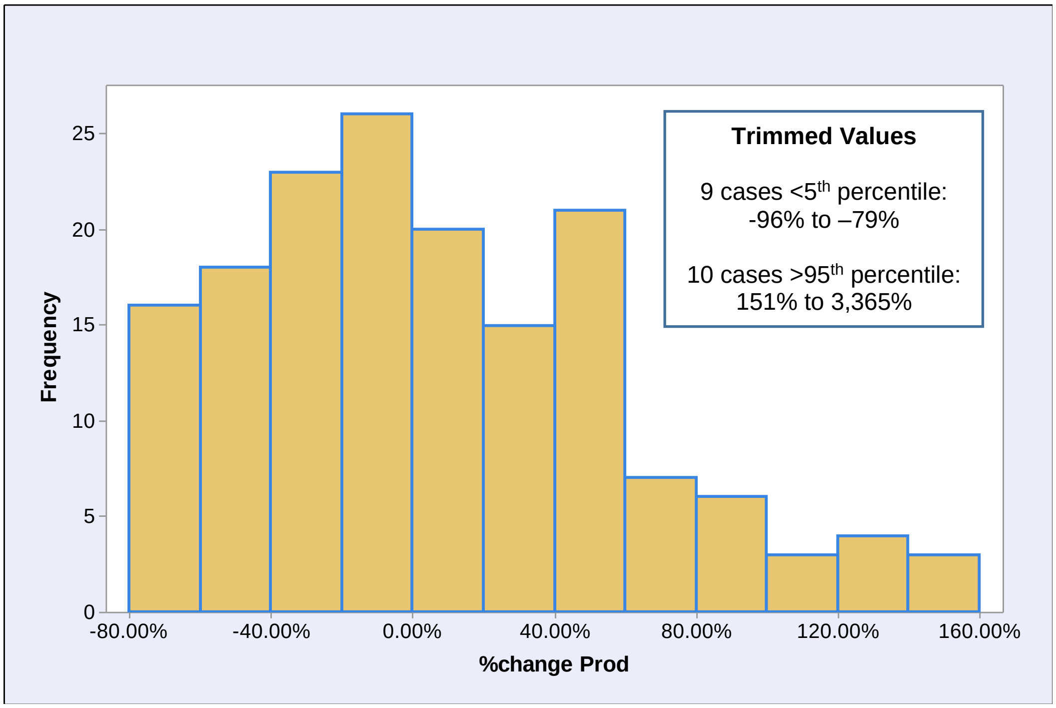


Figure 40: Percentage Difference in Actual versus Estimated Productivity (ESLOC/PM)

Productivity is a question often raised in comparing software development projects. We define productivity as the amount of ESLOC produced per person-month. Additionally, we use 152 hours per person-month14.

Productivity shows a -1% median change across all projects between initial and final SRDRs (Table 3). The mean change was 7%. To varying degrees, 83 cases overestimated their productivity and 79 cases were underestimated. When the projects were grouped into negative and positive productivity groups, the negative group had a -31% median change and the positive group had a 44% increase. Recall that the positive productivity group increased in productivity between initial and final SRDR when requirements, software size, and duration increased.

When projects were grouped by service, the productivity differences were small. The grouping shown in Figure 41 illustrates that there is also no statistical distinction overall between the super-domains, given the relatively large amount of variation in each group. The median changes for AIS, ENG, and RT were 20%, -20%, and -3%, respectively.

[14] See Appendix G; Burden Labor Rate

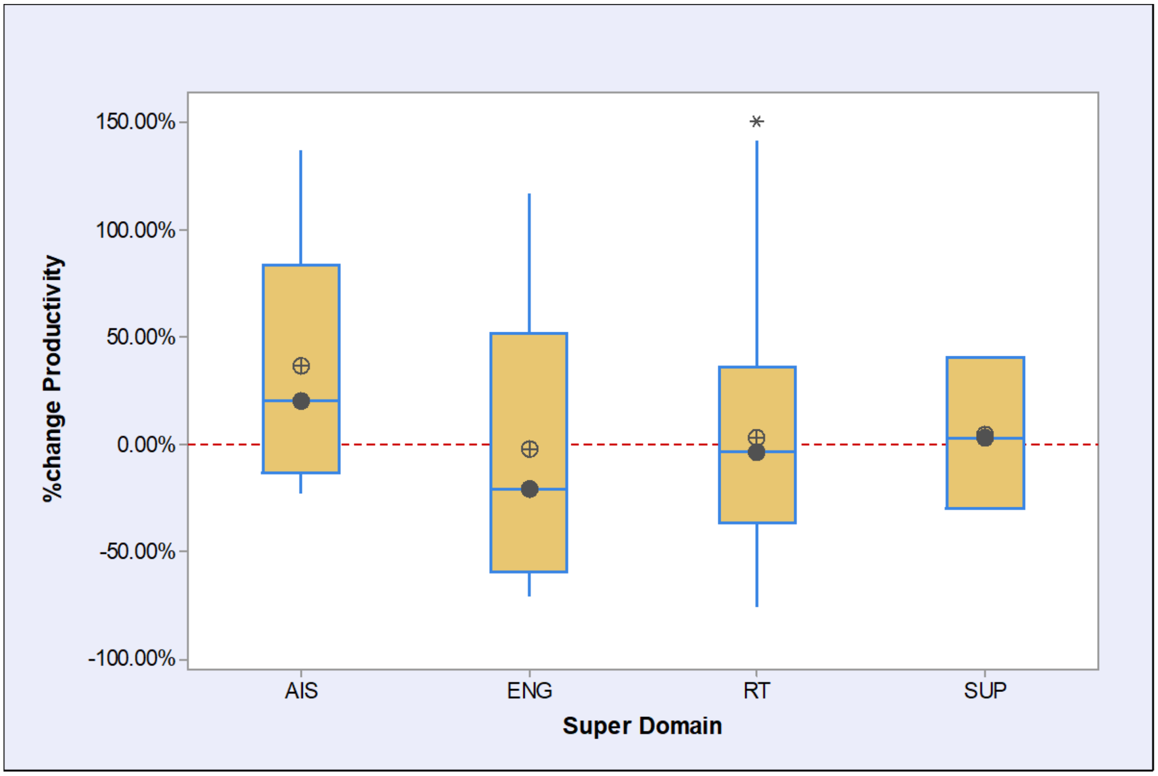


Figure 41: Percentage Change for Productivity by Super-Domain

There are several interesting aspects about the productivity data. First, the overall model for predicting the final productivity using the initial estimate is only moderate. If we factor in the super domain, we derive statistically significant models for AIS, ENG, and RT systems, also of moderate predictive strength. These models are presented following the overall model.

Second, when the data is divided into those cases whose productivity was underestimated (an increase in productivity compared to the initial estimate) versus overestimated, we derive stronger predictive models. Also, when super domain is included we can derive separate models for AIS, ENG, and RT, although some of these models have a very limited number of cases.[15] This, of course, requires that we have paired initial and final submissions to make such a determination and the usefulness for predicting a new project's productivity is limited to use by analogy. However, if a determination can be made at some point during the software development lifecycle as to whether the productivity was over- or underestimated, then these models can be applied at that time to predict better final estimates.

[15] See Appendix F for the models. Use of a predictive model with a small number of cases is usually not recommended.

The results of the regression models on the transformed data are presented below:

n = 162

Regression Equation:

$$\ln Productivity\_Actual = 1.2212 + 0.7439 \ln Productivity\_Estimated$$

which translates to:

$$Actual\ Productivity = 3.39 * (Estimated\ Productivity)^{.74}$$

The adjusted r^2 equals .55; the model accounts for 55% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate by the initial of 88% at the low end (10 ESLOC/person-months) but an overestimate of -52% productivity at the high end (2,000), with the inflection at about 118.

Table 20: Predicted Values for Actual Productivity (ESLOC/Person-Months)

Initial Productivity Estimate	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
10	19	88%	7	54
20	31	57%	11	88
50	62	25%	23	172
75	84	12%	31	232
100	104	4%	38	286
200	175	-13%	64	479
500	345	-31%	125	954
750	467	-38%	168	1,298
1,000	578	-42%	207	1,616
1,250	682	-45%	243	1,917
1,500	781	-48%	277	2,204
1,750	876	-50%	310	2,482
2,000	968	-52%	341	2,750

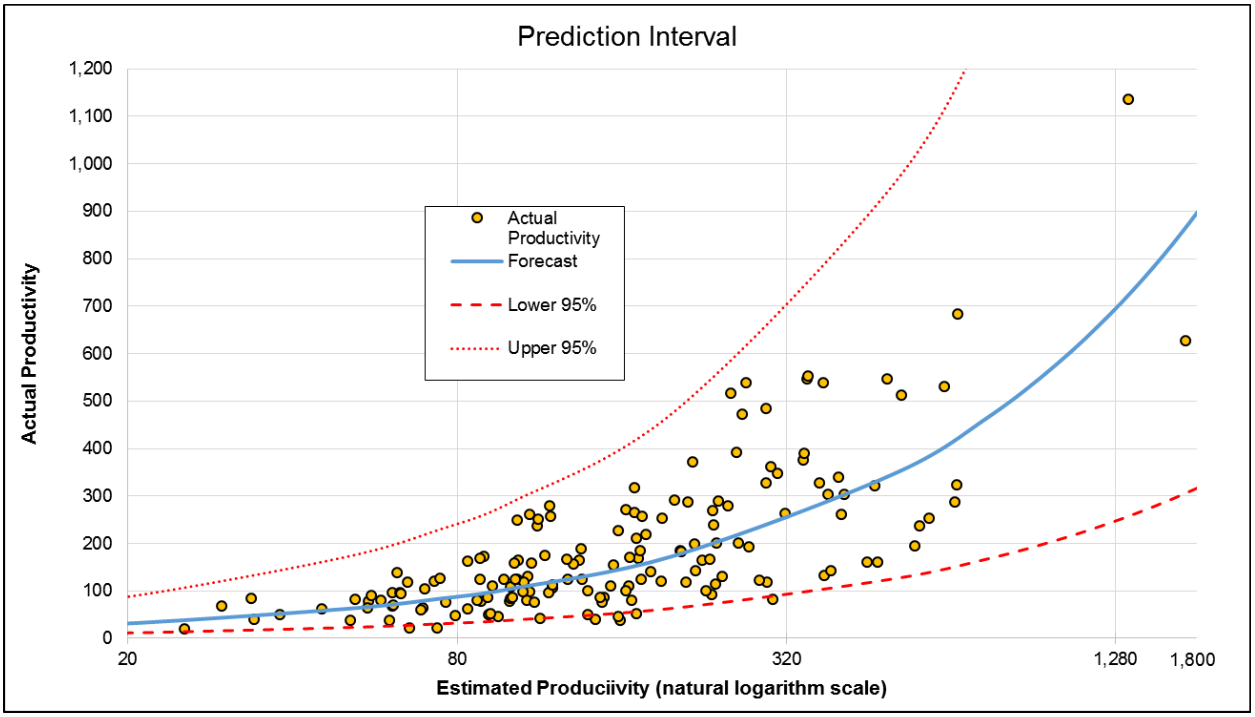


Figure 42: Prediction Interval for Actual Productivity

Given the inflection point is 118, it represents the point at which estimated productivity is statistically most likely to actual productivity. Productivity estimates lower than 118 ESLOC per person month are likely to experience greater productivity. Productivity estimates greater than 118, are likely to experience lower productivity.

118 ESLOC per person month is equal to .77 ESLOC per hour. This is significantly lower than the rule of thumb of 2 SLOC hour used in the 1970's and 1980's.

In practice, estimated productivity is hard to defend. There are several factors which affect realized productivity. As well documented some of the most important influences are related to people (i.e., team cohesion, management effectiveness, etc.). When faced with evaluating productivity estimates, it may be most useful to focus on the area outside the prediction intervals. Anything outside the 95% confidence interval is by definition statistically very unlikely to occur (i.e. dead zone). If a project estimates a productivity outside the confidence interval it warrants further investigation. For example, the largest productivity value in Table 20 is 2750 ESLOC per person month, which equals 18 SLOC per hour. If a project's plan contains a productivity greater than that, it is statistically unlikely to be realized.

AIS – Productivity

$n = 21$

Regression Equation:

$$\mathit{ln\ Productivity\_Actual} = 2.0539 + 0.6651\ \mathit{ln\ Productivity\_Estimated}$$

which translates to:

AIS: $$Actual\ Productivity = 7.80 * (Estimated\ Productivity)^{.67}$$

The adjusted r^2 equals .47; the model accounts for over 47% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate by the initial of 110% at the low end (50) but an overestimate of -13% at the high end (700), with the inflection point at 460.

Table 21: Predicted Values for AIS Actual Productivity (ESLOC/Person-Months)

Initial Productivity Estimate	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
50	105	110%	43	260
75	138	84%	59	320
100	167	67%	74	374
150	218	46%	101	474
200	264	32%	123	568
300	346	15%	160	748
400	419	5%	191	920
500	486	-3%	217	1,088
600	549	-8%	240	1,254
700	608	-13%	261	1,418

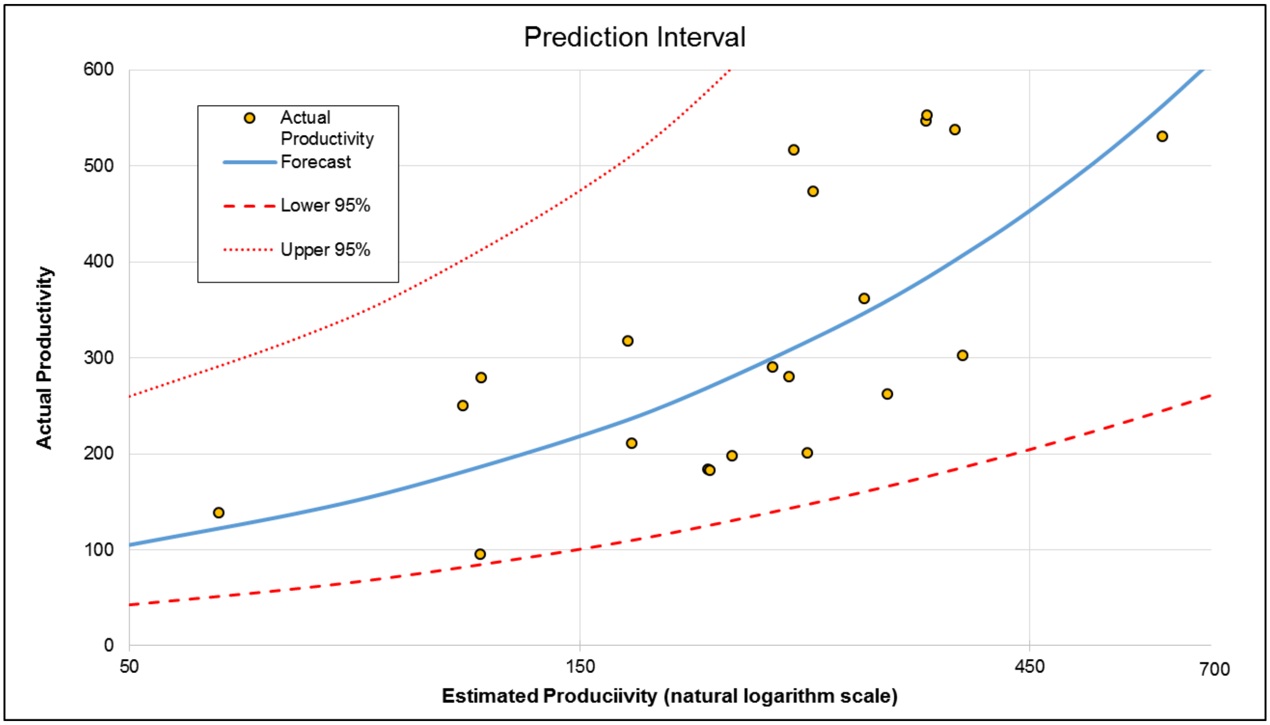


Figure 43: Prediction Interval for AIS Actual Productivity

Given the inflection point is 460, it represents the point at which estimated productivity is statistically most likely to actual productivity. Productivity estimates lower than 460 ESLOC per person month are likely to experience greater productivity. Productivity estimates greater than 460, are likely to experience lower productivity. For AIS, 460 ESLOC per month is equal to 3 ESLOC per hours.

ENG – Productivity

n = 20

Regression Equation:

$$ln\ Productivity\_Actual = 1.5502 + 0.6639\ ln\ Productivity\_Estimated$$

which translates to:

ENG: $$Actual\ Productivity = 4.71 * (Estimated\ Productivity)^{.66}$$

The adjusted r2 equals .57; the model accounts for about 57% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an underestimate of 60% at the low end (50) but show a -62% overestimate at the high end (700), with an inflection point of 100.

Table 22: Predicted Values for ENG Actual Productivity (ESLOC/Person-Months)

Initial Productivity Estimate	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
25	40	60%	10	155

50	63	27%	17	229
75	83	10%	23	292
100	100	0%	29	349
150	131	-13%	38	452
200	159	-21%	46	547
300	208	-31%	60	721
500	292	-42%	82	1,035
750	382	-49%	105	1,393
1,000	462	-54%	124	1,730
1,250	536	-57%	140	2,052
1,500	605	-60%	155	2,363
1,750	670	-62%	168	2,666

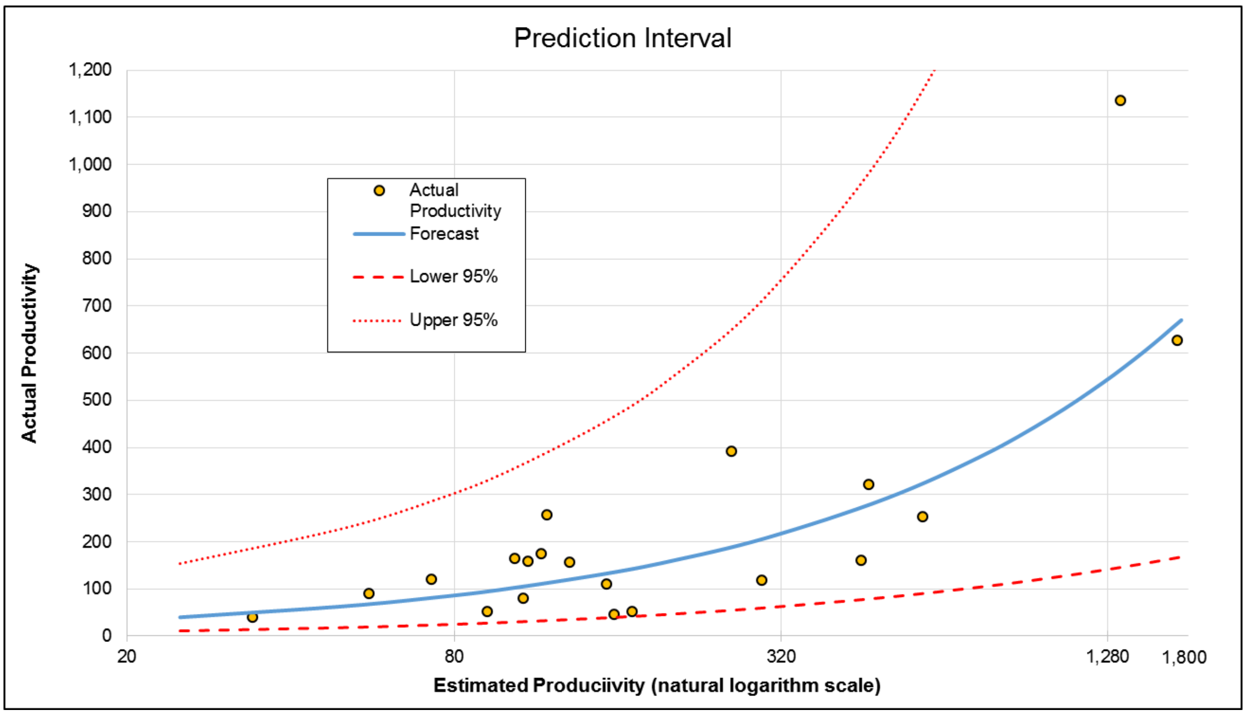


Figure 44: Prediction Interval for ENG Actual Productivity

With an inflection point of 100 for ENG projects. Productivity estimates lower than 100 ESLOC per person month are likely to experience greater productivity. Productivity estimates greater than 100, are likely to experience lower productivity. One hundred ESLOC per person month is equal to approximately 0.7 ESLOC per hour.

RT – Productivity

n = 118

Regression Equation:

$$ln\ Productivity\_Actual = 1.3600 + 0.7027\ ln\ Productivity\_Estimated$$

which translates to:

RT: $$Actual\ Productivity = 3.90 * (Estimated\ Productivity)^{.70}$$

The adjusted r^2 equals .505; the model accounts for over 50% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Predicted values show an overestimate of 97% at the low end (10) but show an underestimate of -44% at the high end (700), with an inflection at 97.

Table 23: Predicted Values for RT Actual Productivity (ESLOC/Person-Months)

Initial Productivity Estimate	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
10	20	97%	7	56
25	37	50%	14	103
50	61	22%	23	164
75	81	8%	30	217
100	99	-1%	37	266
150	132	-12%	49	353
200	161	-19%	60	432
300	215	-28%	80	577
400	263	-34%	97	710
500	307	-39%	113	834
600	349	-42%	128	951
700	389	-44%	142	1,064

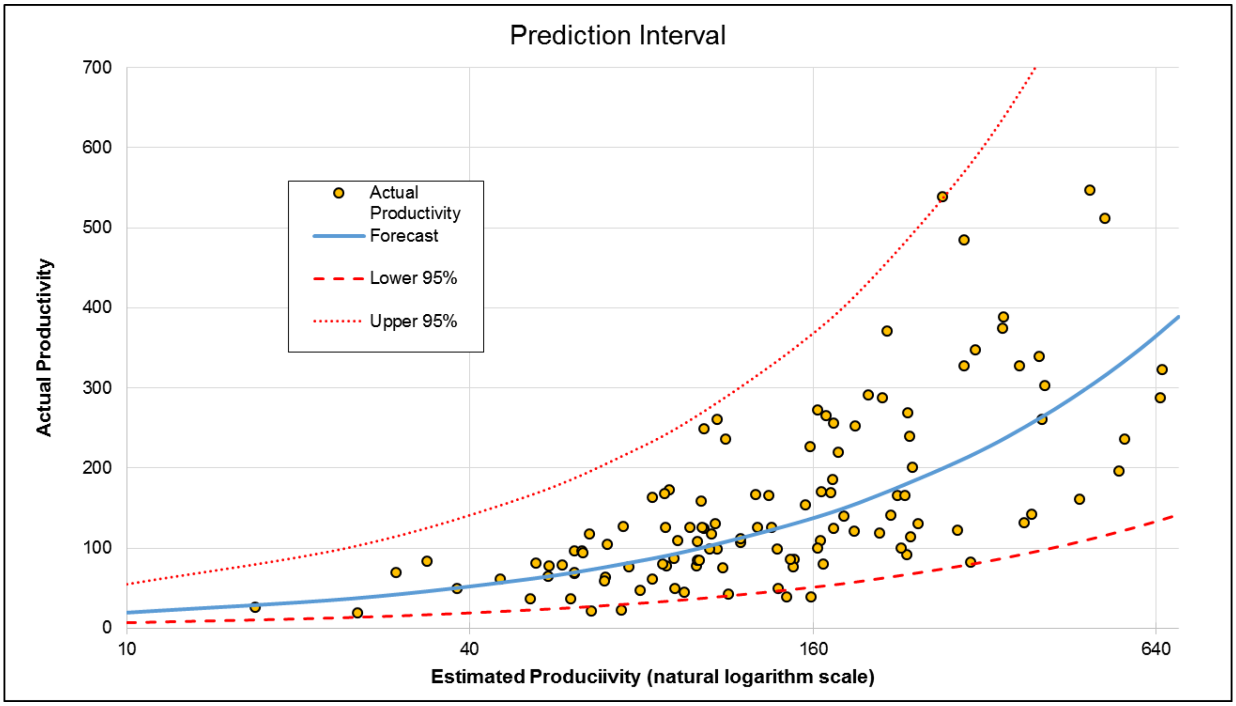


Figure 45: Prediction Interval for RT Actual Productivity

The RT data set yielded the lowest inflection point at 97. Productivity estimates lower than 97 ESLOC per person month are likely to experience greater productivity. Productivity estimates greater than 97, are likely to experience lower productivity. Ninety seven ESLOC per person month is equal to 0.6 ESLOC per hour for RT.

The following models use a subset of the data based on an increase in productivity when comparing the initial estimate to the final outcome, which represents an initial underestimate by the contractor. The first model is for all such cases, followed by separate models for AIS, ENG, and RT.

Positive Productivity

Cases with a Positive Change in Productivity

n = 79

Regression Equation:

$$\textit{ln Productivity\_Actual} = 0.804 + 0.9120 \textit{ ln Productivity\_Estimated}$$

which translates to:

Positive Change: $Actual\ Productivity = 2.2352 * (Estimated\ Productivity)^{.912}$

The adjusted r^2 equals .886; the model accounts for over 88% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. Since this dataset comprises those cases with positive productivity outcomes, the initial submission values will all be under estimates. The data show an underestimate of 83% at the low end (10) and an underestimate of 22% at the high end (1000).

Table 24: Predicted Values for Cases with Underestimated Productivity

Initial Productivity Estimate	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
10	18	83%	11	30
20	34	72%	21	56
50	79	58%	49	127
75	115	53%	72	184
100	149	49%	93	238
200	280	40%	175	449
300	406	35%	253	652
500	647	29%	400	1,046
750	936	25%	574	1,525
1,000	1,217	22%	742	1,995

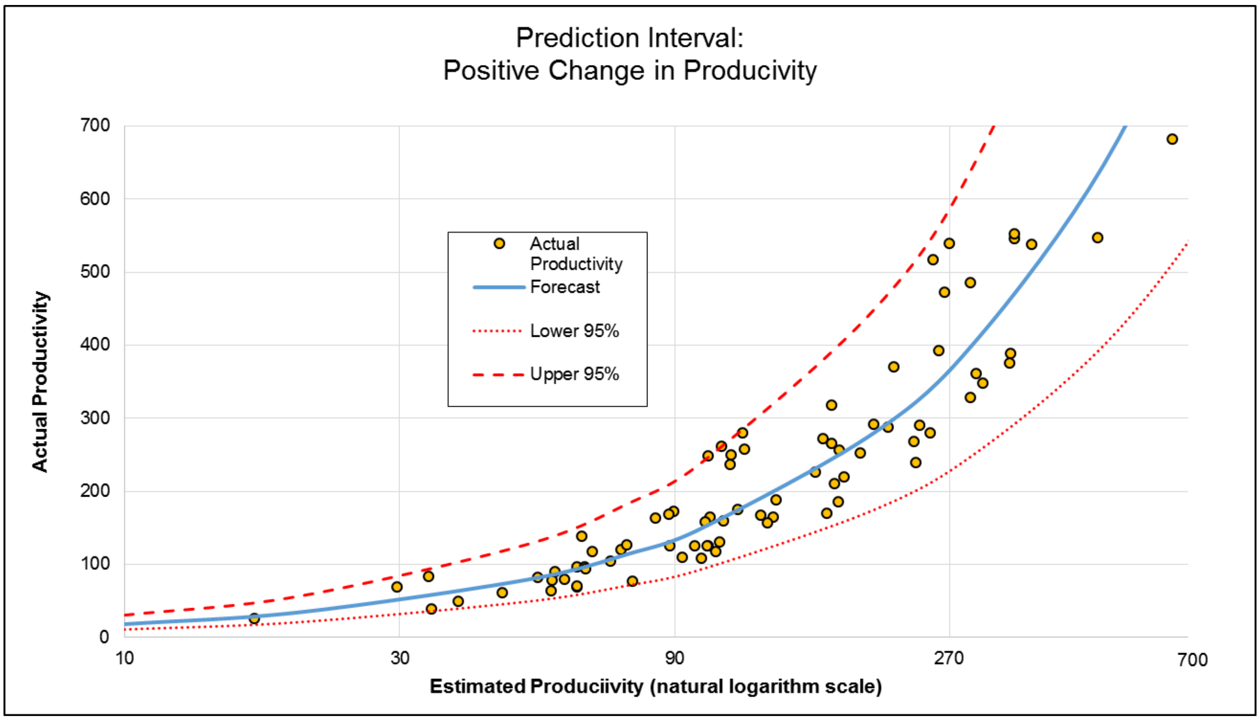


Figure 46: Prediction Interval for Actual Productivity – Cases with Positive Change

In practice, it is essentially impossible to know if a project is going to experience positive or negative productivity. This analysis reveals that once a project is underway and has exhibited positive productivity compared to the initial estimate, then the data can be used to predict the final productivity with far less variance when considering all cases.

AIS – Cases with a Positive Change in Productivity

n = 13

Regression Equation:

$$ln\ Productivity\_Actual = 2.0991 + 0.6983\ ln\ Productivity\_Estimated$$

which translates to:

AIS: $$Actual\ Productivity = 8.1589 * (Estimated\ Productivity)^{.6983}$$

The adjusted r2 equals .761; the model accounts for over 76% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals. For AIS projects that experienced positive productivity gains, the initial submission values will all be under estimates. The data show an underestimate of 209% at the low end (25) and an underestimate of 13% at the high end (700).

Table 25: Predicted Values for Actual Productivity – AIS Cases with Positive Change

Estimated Productivity	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
25	77	209%	37	162
50	125	151%	67	233
75	166	122%	95	292
100	203	103%	119	346
200	330	65%	201	543
300	438	46%	264	727
400	535	34%	316	906
500	626	25%	362	1,082
600	711	18%	402	1,256
700	791	13%	439	1,428

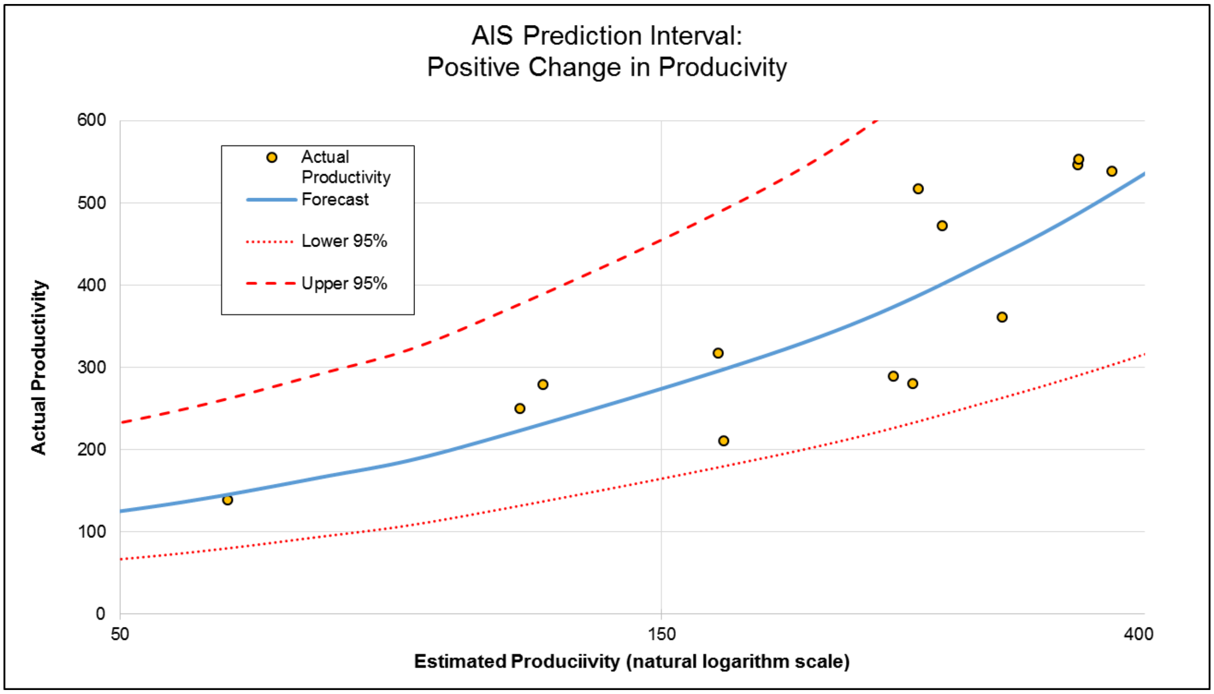


Figure 47: Prediction Interval for Actual Productivity – AIS Cases with Positive Change

Given all positive increases in productivity, the focus is on how much productivity growth is a project likely to experience. Table 25 and Figure 54, show the larger the initial productivity, the less likely huge productivity increase will be realized. For modest estimates (i.e., 25-100 ESLOC per person month), positive productivity gains over a 100% are statistically feasible. More significant forecasts (i.e., 500-700 ESLOC per person month) are statistically likely to experience 25% or less growth in productivity.

ENG – Cases with a Positive Change in Productivity

n = 9

Regression Equation:

$$\ln Productivity\_Actual = 0.0302 + 1.0848 \ln Productivity\_Estimated$$

which translates to:

ENG: $$Actual\ Productivity = 1.0307 * (Estimated\ Productivity)^{1.0848}$$

The adjusted r^2 equals .924; the model accounts for over 92% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals.

Table 26: Predicted Values for Actual Productivity – ENG Cases with Positive Change

Estimated Productivity	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
10	13	25%	6	28
25	34	35%	18	62

50	72	44%	43	120
75	111	49%	69	180
100	152	52%	95	245
150	236	58%	145	387
200	323	62%	193	542
250	412	65%	239	710
300	502	67%	283	888
400	685	71%	369	1,273

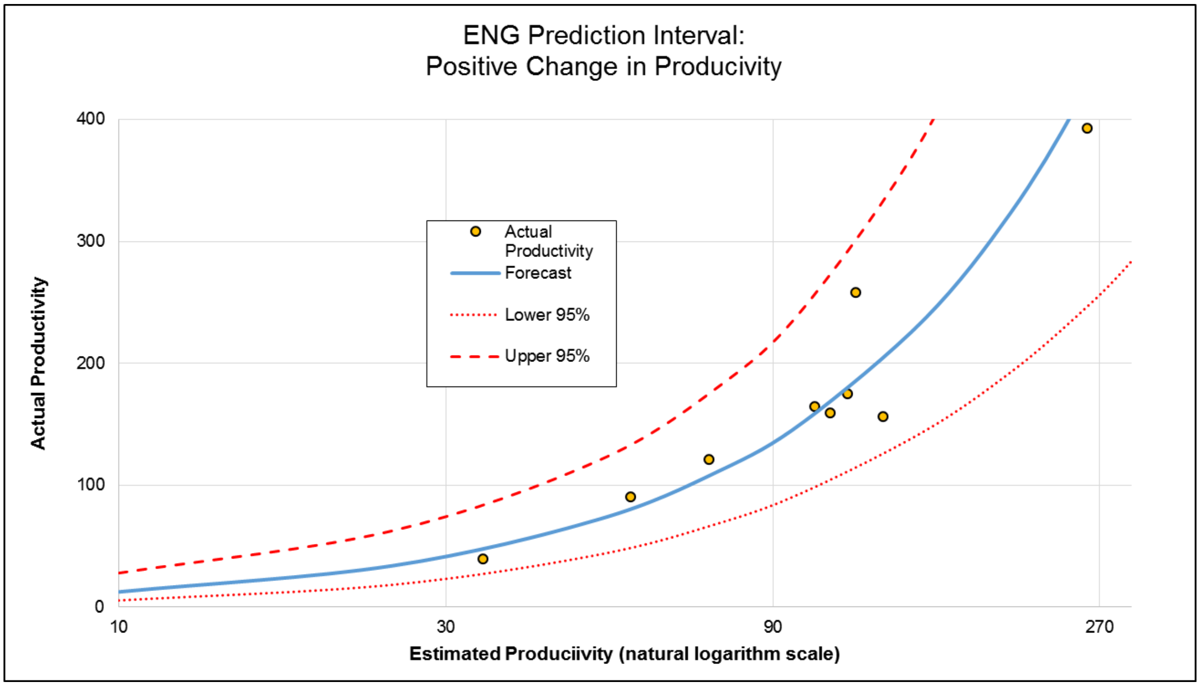


Figure 48: Prediction Interval for Actual Productivity – ENG Cases with Positive Change

Although the analysis was conducted with only 9 data points, the resulting strength in the variance is significant. Given all positive increases in productivity, the focus is on how much productivity growth is a project likely to experience. Table 26 and Figure 48 show productivity increases as initial productivity estimates increase. ENG productivity forecasts are statistically likely to experience a 25% -71% increase in productivity.

RT – Cases with a Positive Change in Productivity

n = 55

Regression Equation:

$$ln\ Productivity\_Actual = 0.8851 + 0.8873\ ln\ Productivity\_Estimated$$

which translates to:

RT: $$Actual\ Productivity = 2.4233 * (Estimated\ Productivity)^{.8873}$$

The adjusted r² equals .878; the model accounts for over 872% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals.

Table 27: Predicted Values for Actual Productivity – RT Cases with Positive Change

Estimated Productivity	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
10	19	**90%**	11	31
15	27	**80%**	16	44
20	35	**75%**	21	56
30	50	**67%**	31	80
50	78	**56%**	49	125
75	112	**49%**	70	178
100	144	**44%**	91	229
200	267	**34%**	167	425
500	601	**20%**	371	975
600	707	**18%**	434	1,152

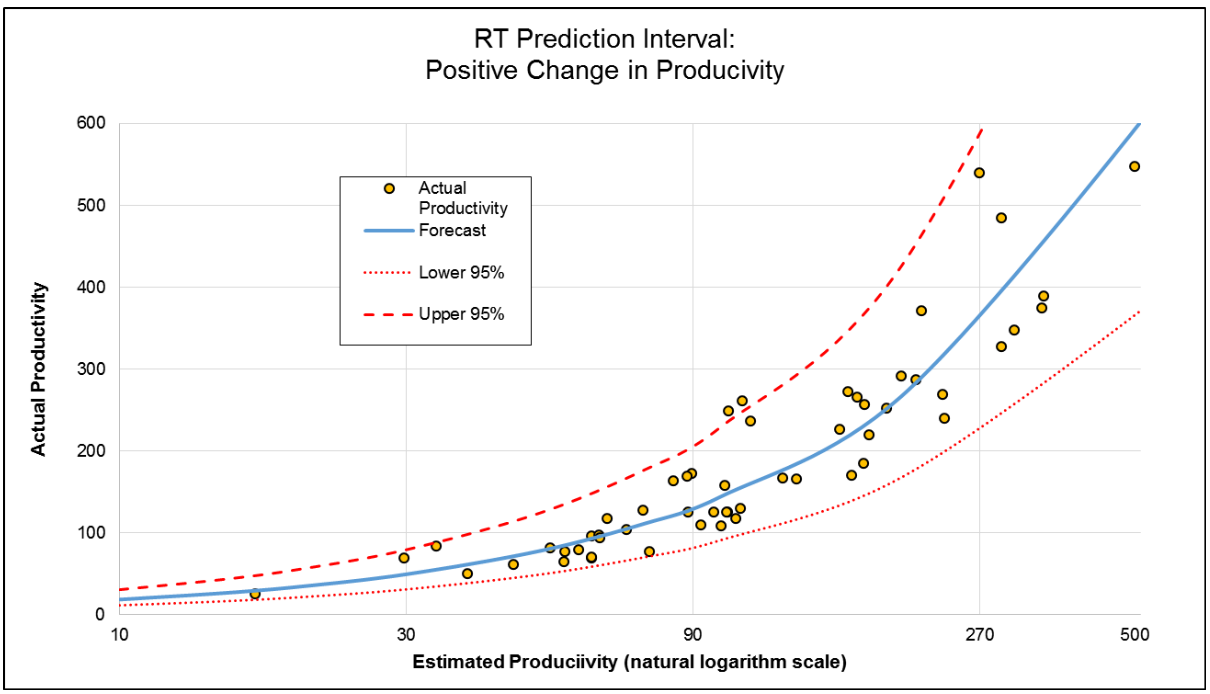


Figure 49: Prediction Interval for Actual Productivity – RT Cases with Positive Change

Given all positive increases in productivity, the focus is on how much productivity growth is a project likely to experience. Table 27 and Figure 49 show the larger the initial productivity, the less likely a huge productivity increase will be realized. For modest estimates (i.e., 10-20 ESLOC per person month), positive productivity gains over 75% are statistically feasible. More significant forecasts (i.e., greater than 75 ESLOC per person month) are statistically likely to experience 50% or less growth in productivity.

Negative Productivity

The following models use a subset of the data based on a decrease in productivity when comparing the initial estimate to the final outcome, which represents an initial overestimate by the contractor. The first model is for all such cases, followed by separate models for AIS, ENG, and RT.

Cases with a Negative Change in Productivity

$n = 83$

Regression Equation:

$$\mathit{ln\ Productivity\_Actual = 0.077 + 0.8910\ ln\ Productivity\_Estimated}$$

which translates to:

$$Actual\ Productivity = 1.0802 * (Estimated\ Productivity)^{.891}$$

The adjusted r^2 equals .758; the model accounts for over 75% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals.

Table 28: Predicted Values for Actual Productivity – All Cases with Negative Change

Estimated Productivity	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
10	8	**80%**	4	19
20	16	**80%**	7	35
50	35	**70%**	16	76
75	51	**68%**	24	108
100	65	**65%**	31	140
200	121	**61%**	57	258
500	274	**55%**	128	588
750	394	**53%**	182	851
1,000	509	**51%**	234	1,107
1,250	621	**50%**	283	1,359
1,500	730	**49%**	332	1,608
1,750	838	**48%**	378	1,855
2,000	944	**47%**	424	2,099

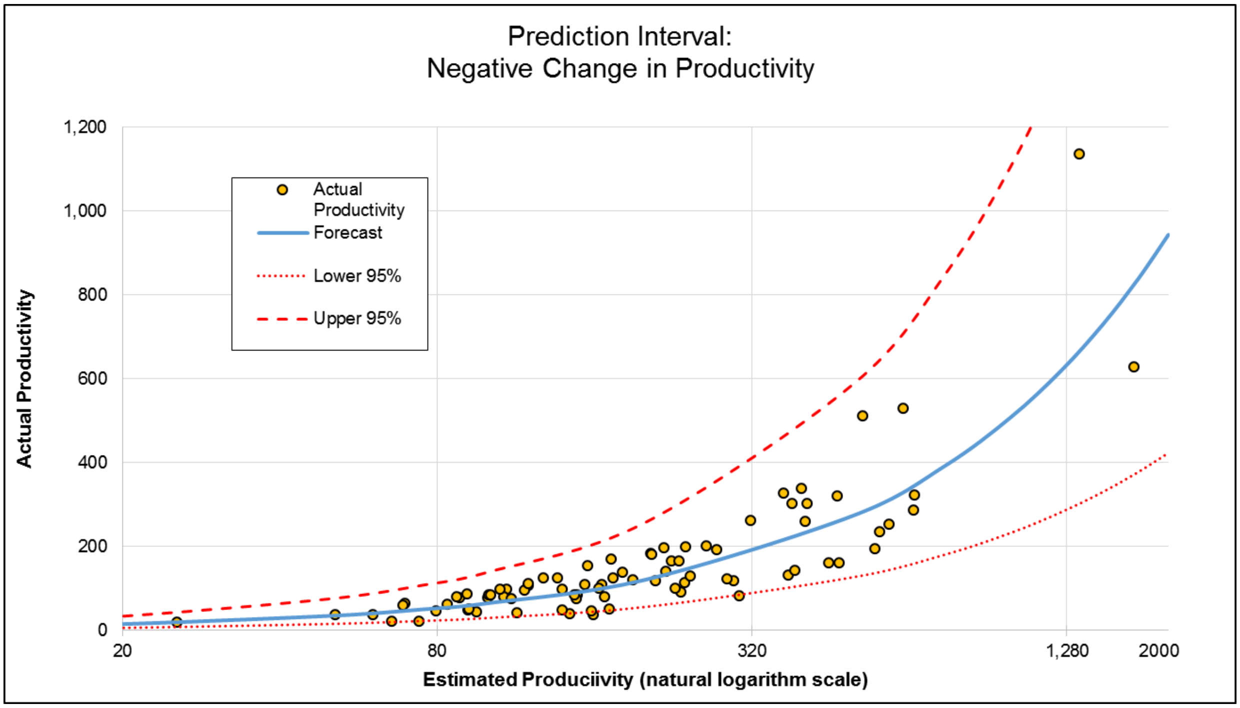


Figure 50: Prediction Interval for Actual Productivity – Cases with Negative Change

As stated earlier, it is essentially impossible to know if a project is going to experience positive or negative productivity. What this analysis reveals is that once a project is underway and has exhibited negative productivity compared with the initial estimate, then this analysis can be used to predict the final productivity with far less variance when considering all cases.

AIS: Cases with a Negative Change in Productivity

n = 8

Regression Equation:

ln Productivity_Actual = -0.078 + 0.9832 ln Productivity_Estimated

which translates to:

AIS: $Actual\ Productivity = 0.9254 * (Estimated\ Productivity)^{.9832}$

The adjusted r^2 equals.982; the model accounts for over 98% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals.

Table 29: Prediction Interval for Actual Productivity – AIS Cases with Negative Change

Estimated Productivity	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
100	86	**86%**	70	105
150	128	**85%**	106	153
200	169	**85%**	142	202
250	211	**84%**	178	250
300	252	**84%**	212	300
350	294	**84%**	247	350
400	335	**84%**	280	400
450	376	**84%**	313	452
500	417	**83%**	345	504
600	499	**83%**	409	609
700	580	**83%**	471	716

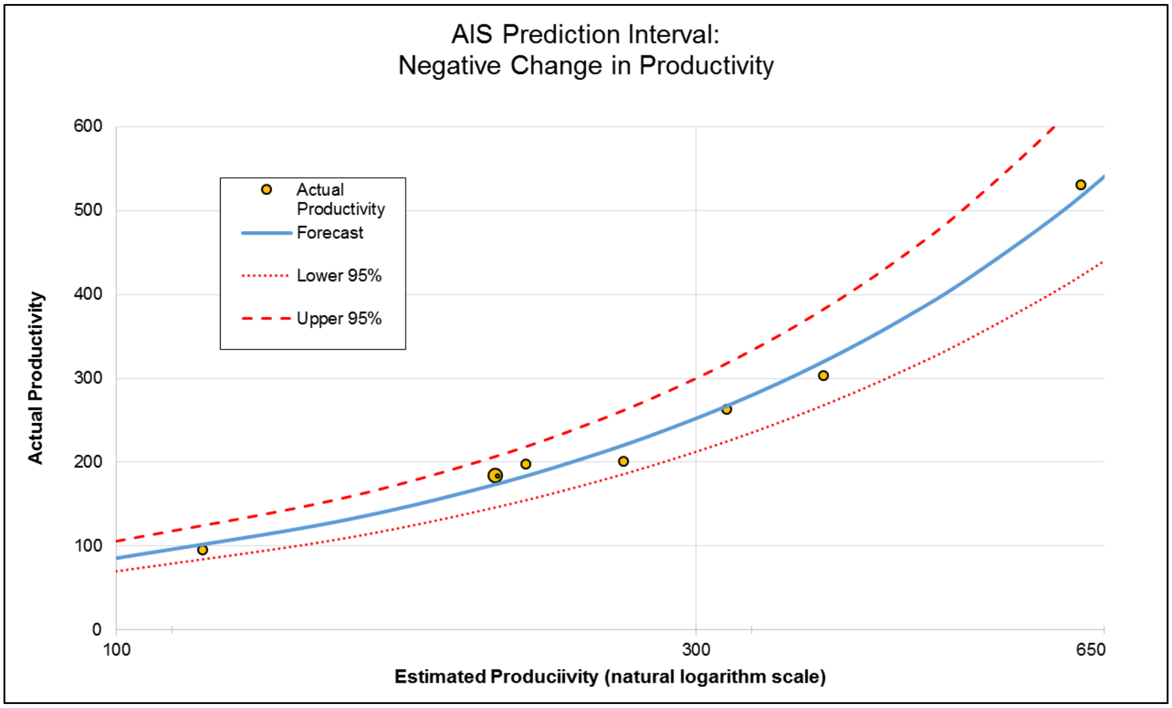


Figure 51: Prediction Interval for Actual Productivity – AIS Cases with Negative Change

Given all negative decreases in productivity, the focus is on how much productivity loss is a project likely to experience. Table 29 and Figure 51 show a decrease between 83% and 86% across the range of estimated productivity values, however with only 8 data points, it is judicious to validate this result against local historical data.

ENG: Cases with a Negative Change in Productivity

n = 11

Regression Equation:

$$\ln Productivity\_Actual = -0.693 + 0.9958 \ln Productivity\_Estimated$$

which translates to:

ENG: $$Actual\ Productivity = 0.5001 * (Estimated\ Productivity)^{.9958}$$

The adjusted r^2 equals .852; the model accounts for over 85% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals.

Table 30: Prediction Interval for Actual Productivity – ENG Cases with Negative Change

Estimated Productivity	**Forecast Productivity**	**Percent difference from Estimate**	**Prediction Interval**	
			Lower 95%	**Upper 95%**
50	25	**50%**	8	74
100	49	**49%**	18	136
150	73	**49%**	27	197
200	98	**49%**	37	258
300	146	**49%**	56	382
500	244	**49%**	93	642
750	365	**49%**	135	983
1,000	486	**49%**	176	1,343
1,250	607	**49%**	214	1,717
1,500	728	**49%**	252	2,104
1,750	848	**48%**	287	2,504

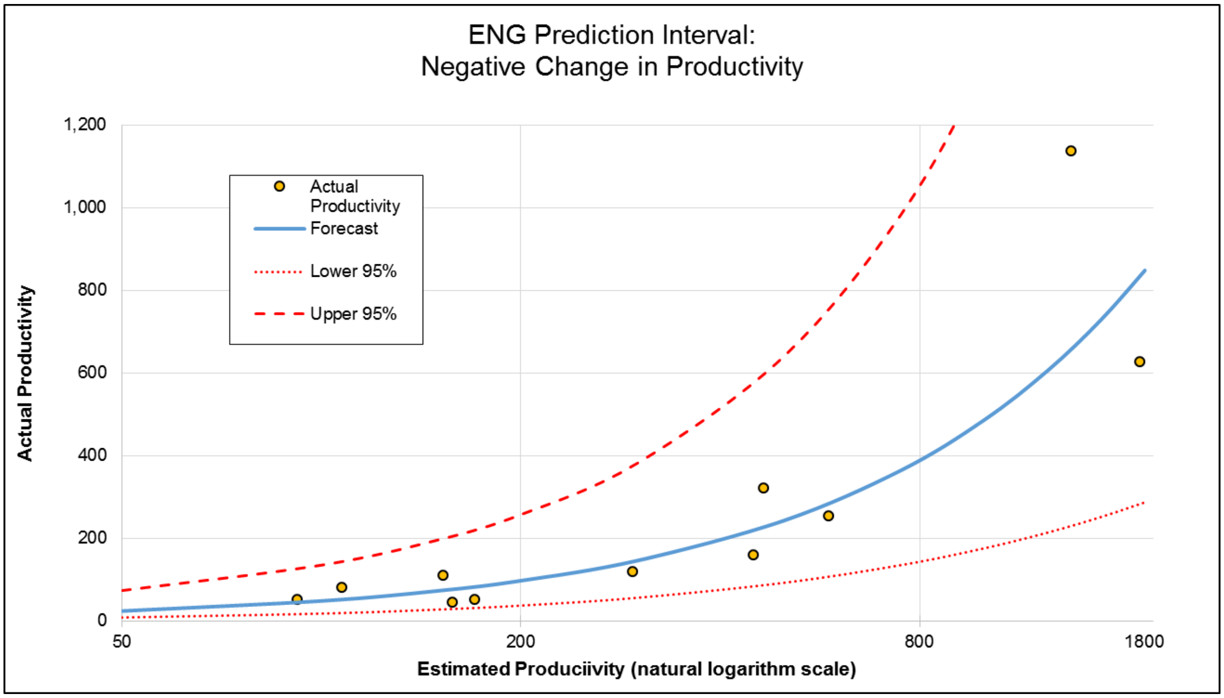


Figure 52: Prediction Interval for Actual Productivity – ENG Cases with Negative Change

Given all negative decreases in productivity, the focus is on how much productivity loss is a project likely to experience. Table 30 and Figure 52 show a decrease between 48% and 50% across the range of estimated productivity values with only 11 data points but a strong calculated variance; it is judicious to validate this result against local historical data.

RT: Cases with a Negative Change in Productivity

n = 63

Regression Equation:

$$ln\ Productivity\_Actual = 0.302 + 0.8431\ ln\ Productivity\_Estimated$$

which translates to:

RT: $Actual\ Productivity = 1.3529 * (Estimated\ Productivity)^{.8431}$

The adjusted r^2 equals .704; the model accounts for over 70% of the variance. Below are the predicted (forecast) values and prediction ranges for a set of new given inputs, followed by a graphic showing the actual data fitted to the model along with the associated prediction intervals.

Table 31: Prediction Interval for Actual Productivity – RT Cases with Negative Change

Estimated Productivity	Forecast Productivity	Percent difference from Estimate	Prediction Interval	
			Lower 95%	Upper 95%
10	9	**90%**	4	22
20	17	**85%**	8	38
50	37	**74%**	17	79
75	52	**69%**	24	110
100	66	**66%**	31	139
150	92	**61%**	44	196
200	118	**59%**	56	250
250	142	**57%**	67	302
300	166	**55%**	78	353
400	211	**53%**	99	452
500	255	**51%**	119	549
600	298	**50%**	138	643
700	339	**48%**	156	736

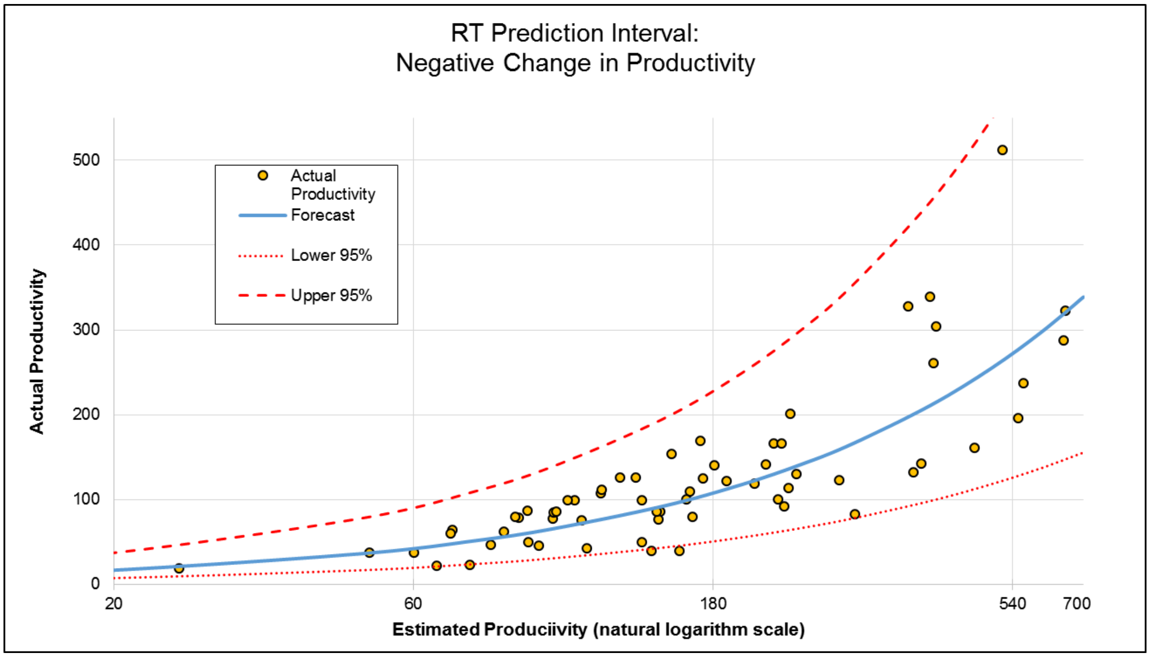


Figure 53: Prediction Interval for Actual Productivity – RT Cases with Negative Change

Given all negative decreases in productivity, the focus is on how much productivity growth is a project likely to experience. Table 31 and Figure 60, show the larger the initial productivity, the less likely a huge productivity decrease will be realized. For modest estimates (i.e., 10-50 ESLOC per person month), negative productivity loss over a 75% are statistically likely. More significant forecasts (i.e., greater than 600 ESLOC per person month) are statistically likely to experience 50% decrease productivity.

5.2.8 Software Growth Summary

Based on historical MDAP/MAIS SRDR data transformed to natural logarithms, we can predict (with a known degree of certainty) the expected outcomes for software size, schedule, and effort. The models presented enable predictions of final outcomes based on initial estimates for MDAP/MAIS programs. Each of the models can be used to construct outcome prediction intervals for any given initial value, although we caution against using the model outside the bounds indicated by the 5th and 95th percentiles for each variable.

To summarize, here are the strongest models to emerge from this analysis:

Requirements	$(r^2 = .936)\ Actual\ Total\ Reqts = 1.2838 * (Estimated\ Total\ Reqts)^{.9456}$
ESLOC	$(r^2 = .849)\ Actual\ Total\ ESLOC = 2.0157 * (Estimated\ ESLOC)^{.964}$
Schedule	$(r^2 = .776)\ Actual\ Total\ Duration = 2.3054 * (Estimated\ Total\ Duration)^{.7878}$
Effort	$(r^2 = .898)\ Actual\ Total\ Hours = 3.3128(Estimated\ Total\ Hours)^{.9097}$

Predicting productivity is less strong unless we separate the underestimated cases from the overestimated cases, which then yield very strong models (r^2 equals .886 and .758, respectively). This indicates that if the productivity could be assessed during the development effort, then the actual outcome could be more accurately predicted. If we also account for the type of super domain, these models increase in strength.

Schedule duration can also be separately predicted for the three services. We show that total effort hours can also be predicted by using the initial estimate for ESLOC, although the fit is not as strong (r2 = .674) as using the initial estimate for hours. We also show how the prediction interval becomes tighter when the confidence level for the prediction is reduced.

Perhaps the most useful takeaway from this analysis are the prediction tables. The tables provide the predicted value along with the prediction interval at a 95% confidence level. These can be used in the absence of any available estimates, or as a sanity check against estimates coming from other sources. New values can easily produce a ballpark forecast by interpolation or the actual equation can be used for calculation. The datasets we used are also available for distribution which enable users to reproduce the models with their own statistical software.

As mentioned earlier, no further adjustments were made in case selection once the data were trimmed. Undoubtedly, the models could be improved (and the predictive intervals narrowed) with substantive knowledge concerning the behavior of outliers which could provide meaningful reasons for their exclusion from a model. Also, any additional data supplied during the interim of the project—which is under consideration by the DoD—could further calibrate and improve a model's fit. This would be especially useful in the productivity models where the best fits were determined by whether the original submission over- or underestimated the productivity. A midcourse determination of productivity would then indicate which sub-model was appropriate to estimate the final productivity for the project.

6 Conclusions and Next Steps

This analysis shows that the cost of software development varies depending on several factors. The class or super-domain of software makes a difference in the cost of software. Different super domains have different levels of difficulty that cause more effort to be expended on more difficult software. On an average-size project, AIS software costs $31,350 a month and RT software costs $101,250 a month—more than three times as much.

The time to develop software also drives cost. Based on an average-size project, shorter duration projects cost disproportionately more than longer duration projects. It was shown that team size is clearly NOT determined solely by the size of the software to be built.

The performance of a project also drives cost. The analysis looked at best, average, and worst performing projects within each super-domain. Unfortunately there was not enough background data on projects to investigate why best and worst projects perform differently. This leads to the next steps.

There is an effort to link the project data back to source documents and other data to provide the capability to investigate the data more fully. There is a lot of data and source material, and some progress has been made to date with a lot more to do.

There is additional SRDR data that can be added to this analysis, and new data is submitted every quarter. More data would increase the fidelity of grouping the data into different super-domains of software, providing a more robust analysis.

The intent of this report is to provide a characterization of the Department of Defense software portfolio and to demonstrate how the SRDR data is useful in gaining insights into software development costs. More analysis can be done, but what we want to know from you is, “What are the important questions that need answers?” The authors wish to receive feedback on this report and input for useful extensions. For comments and suggestions, please contact:

fact-book@sei.cmu.edu

Appendix A: Acronyms and Definitions

AIS	Automated Information System Software. See Appendix C: Super-domains.
DACIMS	Defense Automated Cost Information Management System
ENG	Engineering Software. See Appendix C: Super-domains.
ESLOC	Equivalent source lines of code. See Appendix B: Equivalent Source Lines of Code.
FTE	Full-time equivalent; the number of total hours worked divided by the maximum number of compensable hours in a full-time schedule. For example, if the normal schedule for a quarter is defined as 35 hours per week * (52 weeks per year / 4), 411.25 hours, then someone working 100 hours during that quarter represents 100/411.25 = 0.24 FTE.
KESLOC	Thousands (K) of ESLOC
Ln	Natural log
MAIS	Major Automated Information System
MDAP	Major Defense Acquisition Program
MS	Mission-Support Software. See Appendix C: Super-domains.
OpEnv	Operating environment. See Appendix D: Operating Environment.
PD	Person days; a measure of effort based on 8 hours per day for requirements through final qualification testing activities; 1 PD = 1 calendar day only when one person is working on the project.
PM	Person months; a measure of effort based on an average of 152 labor hours in a month. The average includes vacation time, sick time, and holidays.
Project Data	Data from an SRDR product event
RT	Real Time Software systems. See Appendix C: Super-domains.
SD	Standard deviation; the amount of variation about the mean value of a measure. ±1 standard deviation covers about 67% of projects
SE	Standard error; a measure of the accuracy of the predictions from a regression model.
SRDR	Software Resources Data Report

Appendix B: Equivalent Source Lines of Code (ESLOC)

This analysis uses a product-size measure based on software source lines of code (SLOC). A key issue in using SLOC as a measure of work effort and duration is the difference in work required to incorporate software from different sources:

- new code
- modified code (changed in some way to make it suitable)
- reused code (used without changes)
- auto-generated code (created from a tool and used without change)

Each of these computer code sources requires a different amount of work effort to incorporate into a software product. The challenge is in coming up with a single measure that includes all of the code sources.

The approach taken here is to normalize all code sources to the *equivalent* of a new line of code. This is done by taking a portion of the measures for modified, reused, and auto-generated code. The portioning is based on the percentage of modification to the code based on changes to the design, code and unit test, and integration and test documents. This approach is adopted from the COCOMO II Software Cost Estimation Model.[16]

Equivalent source lines of code (ESLOC), then, is the homogeneous sum of the different code sources. The portion of each code source is determined using a formula called an Adaptation Adjustment Factor (AAF):

$$AAF = (0.4 \times \%DM) + (0.3 \times \%CM) + (0.3 \times \%IM)$$

Where

%DM: Percentage Design Modified

%CM: Percentage Code and Unit Test Modified

%IM: Percentage Integration and Test Modified

Using a different set of percentages for the different code sources, ESLOC is expressed as

$$\begin{aligned} ESLOC = {} & \text{New SLOC} + \\ & (AAF_M \times \text{Modified SLOC}) + \\ & (AAF_R \times \text{Reused SLOC}) + \\ & (AAF_{AG} \times \text{Auto-Generated SLOC}) \end{aligned}$$

New code does not require any adaptation parameters, since nothing has been modified.

Auto-generated code does not require the DM or CM adaptation parameters. However, it does require testing, IM. If auto-generated code does require modification, then it becomes modified code, and the adaptation factors for modified code apply.

16 Boehm, B., Abts, C., Brown, W, Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, R., and Steece, B., *Software Cost Estimation with COCOMO II*, Prentice Hall, 2000, p. 22.

Reused code does not require the DM or CM adaptation parameters, either. It also requires testing, IM. If reused code does require modification, then it becomes modified code and the adaptation factors for modified code apply.

Modified code requires the three parameters, DM, CM, and IM, representing modifications to the modified code design, code, and integration testing.

The equivalent sizes for all of the projects are shown in the next two histogram graphs. The first histogram shows that sizes for the projects do not have a normal distribution. The analyses in this Factbook rely on statistical methods that require a normally distributed dataset.

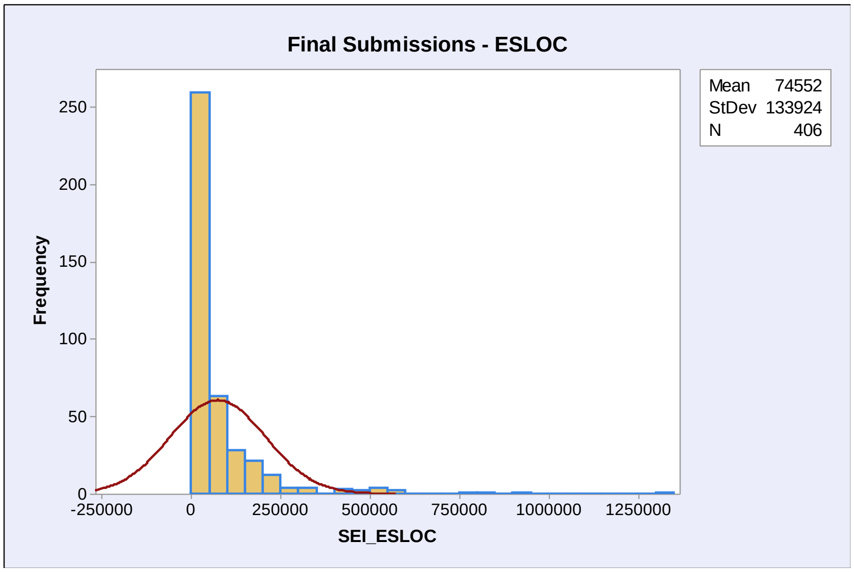


Figure 54: Final Submissions - ESLOC

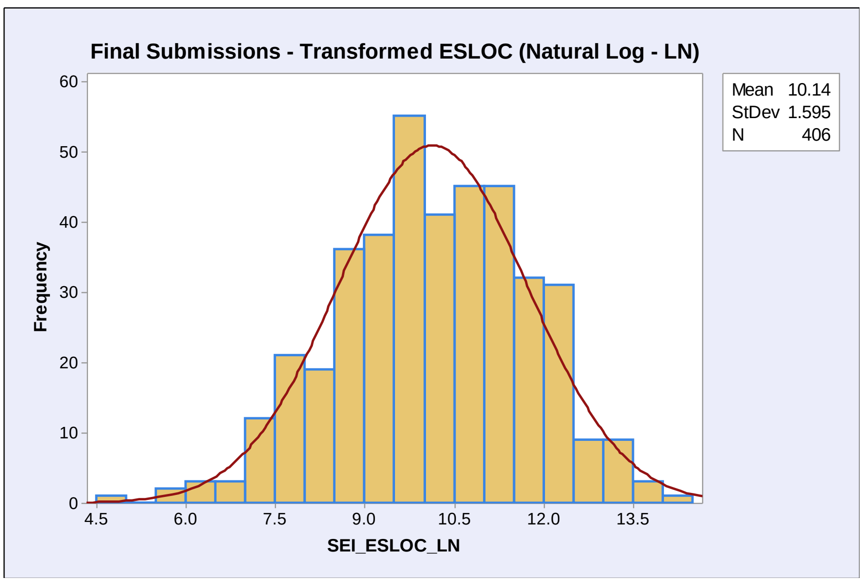


Figure 55: Final Submission - Transformed (LN) ESLOC

Appendix C: Super Domains

Real Time (RT)

Real time is the most complex type of software. These projects take the most time and effort for a given system size due to the lower language levels, high level of abstraction, and increased complexity:

- tightly coupled interfaces
- real time scheduling requirements
- very high reliability requirements (life critical)
- generally severe memory and throughput constraints
- often executed on special-purpose hardware

Examples of software domains in this super-domain are: sensor control and signal processing, vehicle control, vehicle payload, and real time embedded.

Engineering (ENG)

Engineering is a software type of medium complexity.

- multiple interfaces with other systems
- constrained response-time requirement
- high reliability but not life critical
- generally executed on commercial off-the- shelf (COTS) software applications

Examples of software domains in this super-domain are: mission processing, executive, automation and process control, scientific systems, and telecommunications.

Support (SUP)

Support is the least complex type of software.[17] Software is often written in more human-oriented languages and performs common business functions such as order entry, inventory, human resources, financial transactions, and data processing and storage.

- relatively less complex
- self-contained or few interfaces
- less stringent reliability requirement

Examples of software domains in this super-domain are: planning systems, non-embedded training, software tools, and non-embedded test software.

---

17 Because there were so few projects in the SUP domain in our data set, we did not include the SUP domain in the analysis results in this report.

Automated Information Systems (AIS)

AIS is software that automates information processing. These applications allow the designated authority to exercise control over the accomplishment of the mission. Humans manage a dynamic situation and respond to user input in real time to facilitate coordination and cooperation.

Examples of software domains in this super-domain are: intelligence and information systems, software services, and software applications.

Appendix D: Operating Environments

Aerial Vehicle (AV)

Examples of aerial vehicles are

- manned: fixed-wing aircraft, helicopters
- unmanned: remotely piloted air vehicles

Ground Site (GS)

Examples of ground sites are

- fixed: command post, ground operations center, ground terminal, test faculties
- mobile: intelligence-gathering stations mounted on vehicles, mobile missile launcher, handheld devices

Ground Vehicle (GV)

Examples of ground vehicles are

- manned: tanks, howitzers, personnel carrier, mobile missile launcher
- unmanned: robots

Maritime Vessel (MV)

Examples of maritime vessels are

- manned: aircraft carriers, destroyers, supply ships, submarines
- unmanned: mine-hunting systems, towed sonar array

Ordnance Vehicle (OV)

Examples of ordnance vehicles are

- air-to-air missiles, air-to-ground missiles, smart bombs, strategic missiles

Space Vehicle (SV)

Examples of space vehicles are

- manned: passenger vehicle, cargo vehicle, space station
- unmanned: orbiting satellites (weather, communications), exploratory space vehicles

Appendix E: Transforming Data

The data means, standard deviations, and trend lines through data used in this analysis assume that the data has a bell-shaped normal distribution.

For example, the two figures at right show the same data for the number of FTEs. The top chart shows the data skewed up against the left axis with a non-bell-shaped distribution. The data in the bottom chart has been transformed into a near normal distribution by converting the data to their natural log values, i.e., ln(FTE).[18]

The impact of non-normal distribution versus normal distribution in the data for the value of the mean can be seen in these two charts.

- mean, non-normal distribution (top chart): 10.389
- mean, normal distribution (bottom chart): 5.2

The difference between the two means shows that the mean for non-normal data is twice the value for the mean for normal data and is very misleading. Note that the transformed mean is relatively close to the median of the untransformed data. It is always best practice to check the normality assumption of data before reporting the data's parametric statistics.

Throughout this report, the data used for prediction models were transformed to their natural logarithm values ($\log_e = \log_{2.718}$).

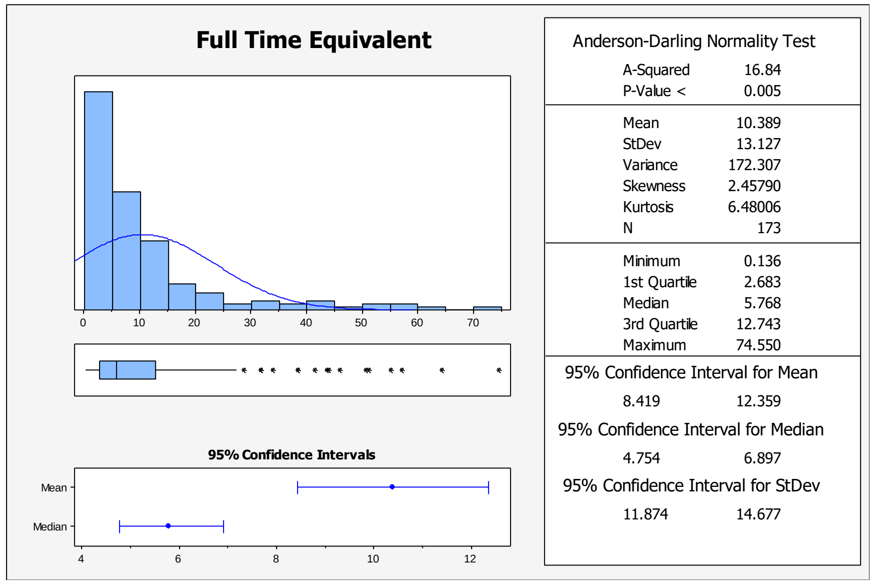


Figure 56: Skewed Distribution of FTE

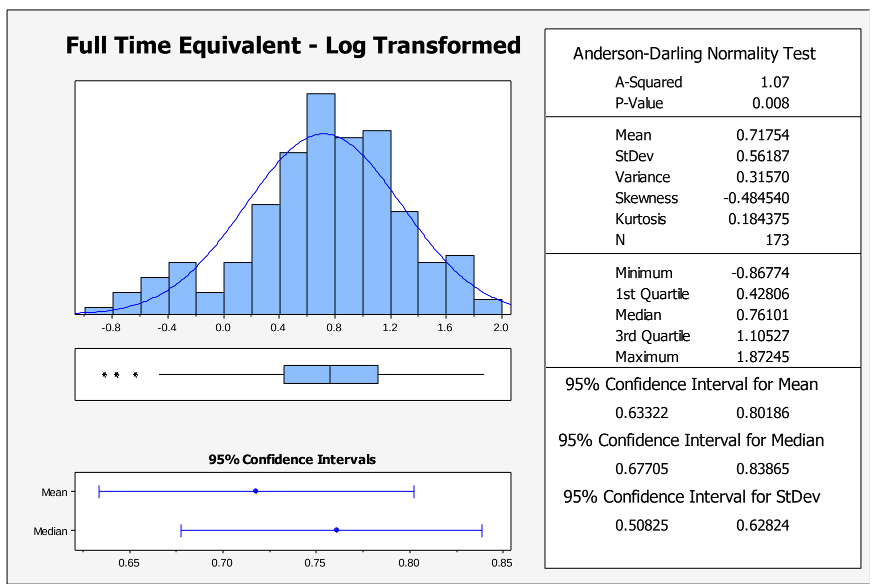


Figure 57: Near Normal Distribution of FTE in Log Values

18 To achieve a more normal distribution of data it is common practice to use a log transformation.Throughout this report we chose to use a natural log transformation for the sake of consistency. The authors felt that a natural log transformation adequately satisfied the assumption of a normal data distribution and its consistent use eased its explanation and interpretation.

Appendix F: Predictive Models

We statistically investigated many variables to establish predictive relationships to the outcome variables (Total Requirements, Total ESLOC, Total Duration, Total Hours, and Productivity). We found that surprisingly little explanatory power was discovered using the difference or percentage change comparing the estimated values to the final values. Instead we found statistically significant relationships using the initial estimates to predict the final outcomes when the data were transformed to their natural logarithm values.

The results of these models are presented and discussed in Section 5. We present the full Minitab statistical output here. Each of these models utilized datasets created by trimming the bottom 5% and the top 5% of the cases based on each variables' percentage change. The resulting spread of values is reported in Section 5.2.

Presented below are the best fitted statistical model outputs.

Total Requirements

Regression Analysis: ln Total Req_Final versus ln Total Reqs_initial

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	330.739	93.65%	330.739	330.739	2153.59	0.000
ln Total Reqs_i_1	1	330.739	93.65%	330.739	330.739	2153.59	0.000
Error	146	22.422	6.35%	22.422	0.154		
Lack-of-Fit	129	20.056	5.68%	20.056	0.155	1.12	0.419
Pure Error	17	2.366	0.67%	2.366	0.139		
Total	147	353.161	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.391887	93.65%	93.61%	23.0106	93.48%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.250	0.122	(0.009, 0.491)	2.05	0.042	
ln Total Reqs_i_1	0.9456	0.0204	(0.9054, 0.9859)	46.41	0.000	1.00

Regression Equation

ln Total Req_F_1 = 0.2498 + 0.9456 ln Total Reqs_i_1

which translates to: $Actual\ Total\ Reqts = 1.2838 * (Estimated\ Total\ Reqts)^{.9456}$

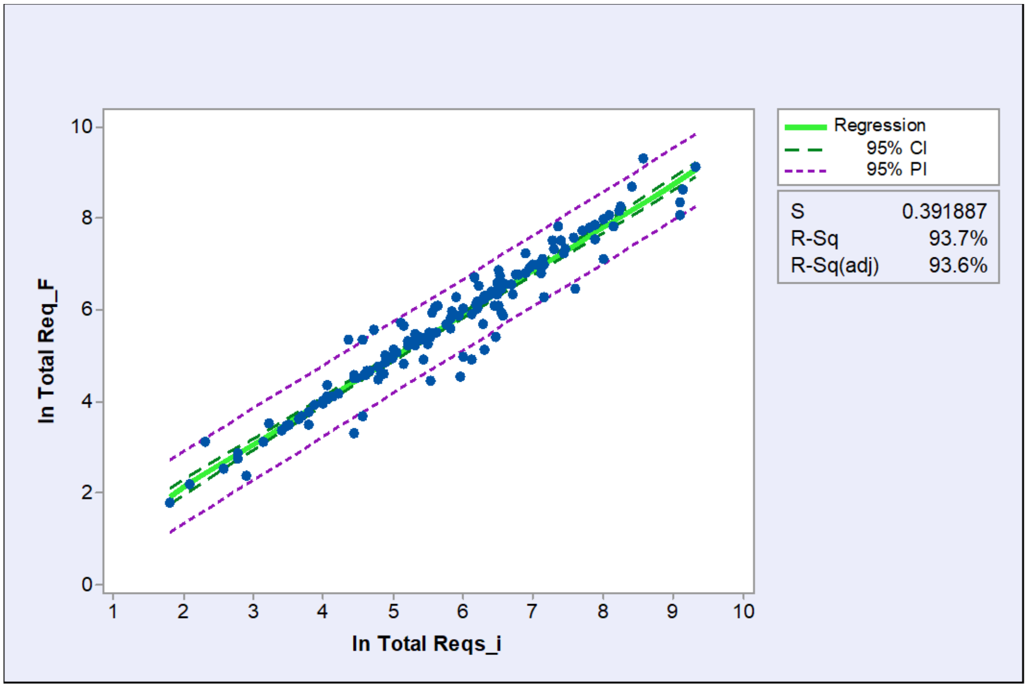


Figure 58: Fitted Regression Plot - Actual Requirements

Total ESLOC

Regression Analysis: ln ESLOC_F versus ln ESLOC_i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	289.99	85.03%	289.99	289.993	908.56	0.000
ln ESLOC_i	1	289.99	85.03%	289.99	289.993	908.56	0.000
Error	160	51.07	14.97%	51.07	0.319		
Total	161	341.06	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.564959	85.03%	84.93%	52.2909	84.67%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.701	0.325	(0.060, 1.342)	2.16	0.032	
ln ESLOC_i	0.9640	0.0320	(0.9008, 1.0271)	30.14	0.000	1.00

Regression Equation

ln ESLOC_F = 0.701 + 0.964 ln ESLOC_i

which translates to: $Actual\ Total\ ESLOC = 2.0157 * (Estimated\ ESLOC)^{.964}$

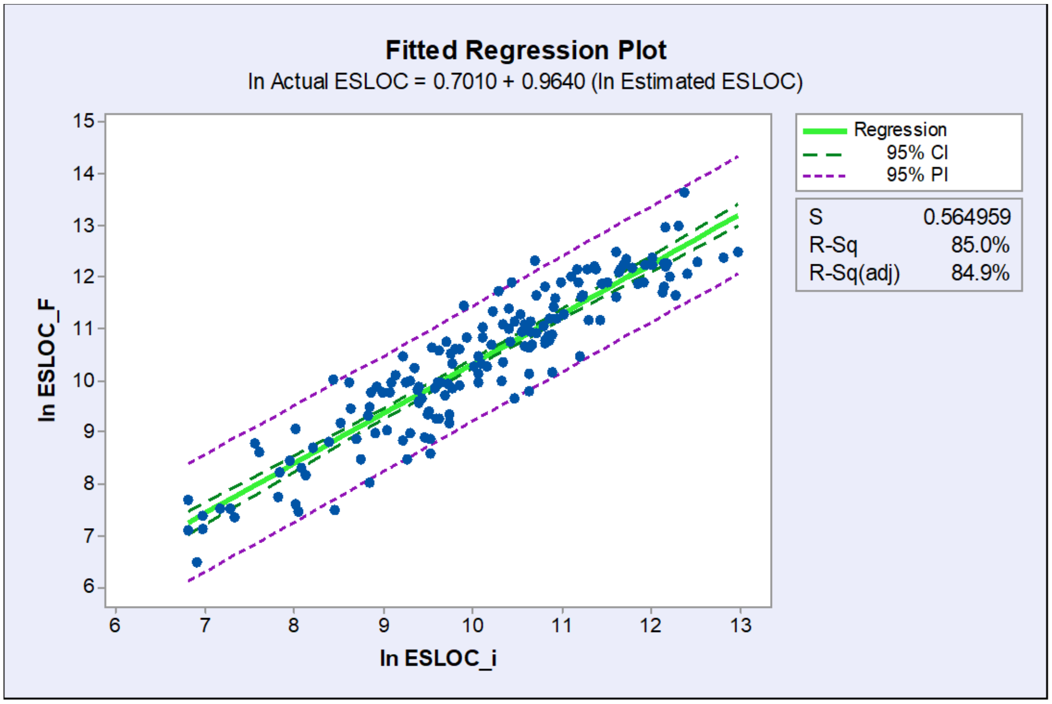


Figure 59: Fitted Regression Plot - Actual ESLOC

Total Schedule Duration

Regression Analysis: ln Months_F versus ln Months_i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	35.928	77.79%	35.928	35.9278	556.89	0.000
ln Mos i	1	35.928	77.79%	35.928	35.9278	556.89	0.000
Error	159	10.258	22.21%	10.258	0.0645		
Lack-of-Fit	95	8.789	19.03%	8.789	0.0925	4.03	0.000
Pure Error	64	1.469	3.18%	1.469	0.0229		
Total	160	46.186	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.253998	77.79%	77.65%	10.6158	77.01%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.835	0.111	(0.615, 1.055)	7.50	0.000	
ln Mos i	0.7878	0.0334	(0.7219, 0.8537)	23.60	0.000	1.00

Regression Equation

```
ln Mos F = 0.8352 + 0.7878 ln Mos i
```

which translates to: $Actual\ Total\ Duration = 2.3054 * (Estimated\ Total\ Duration)^{.7878}$

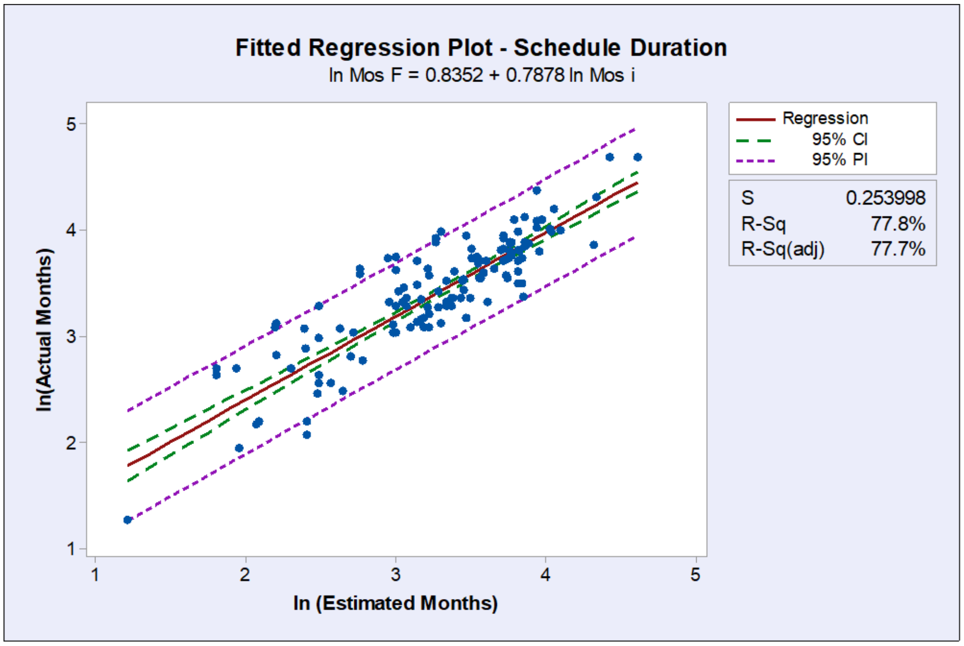


Figure 60: Fitted Regression Plot - Actual Duration

Results for: ARMY Schedule Duration (Subset)

Regression Analysis: ln Months_F versus ln Months_i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	16.0477	83.16%	16.0477	16.0477	311.14	0.000
ln Mos i	1	16.0477	83.16%	16.0477	16.0477	311.14	0.000
Error	63	3.2493	16.84%	3.2493	0.0516		
Lack-of-Fit	35	2.9766	15.42%	2.9766	0.0850	8.73	0.000
Pure Error	28	0.2728	1.41%	0.2728	0.0097		
Total	64	19.2971	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.227105	83.16%	82.89%	3.54211	81.64%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.515	0.162	(0.190, 0.839)	3.17	0.002	
ln Mos i	0.8657	0.0491	(0.7676, 0.9638)	17.64	0.000	1.00

Regression Equation

```
ln Mos F = 0.5146 + 0.8657 ln Mos i
```

which translates to:

ARMY: $Actual\ Total\ Duration = 1.6729 * (Estimated\ Total\ Duration)^{.8657}$

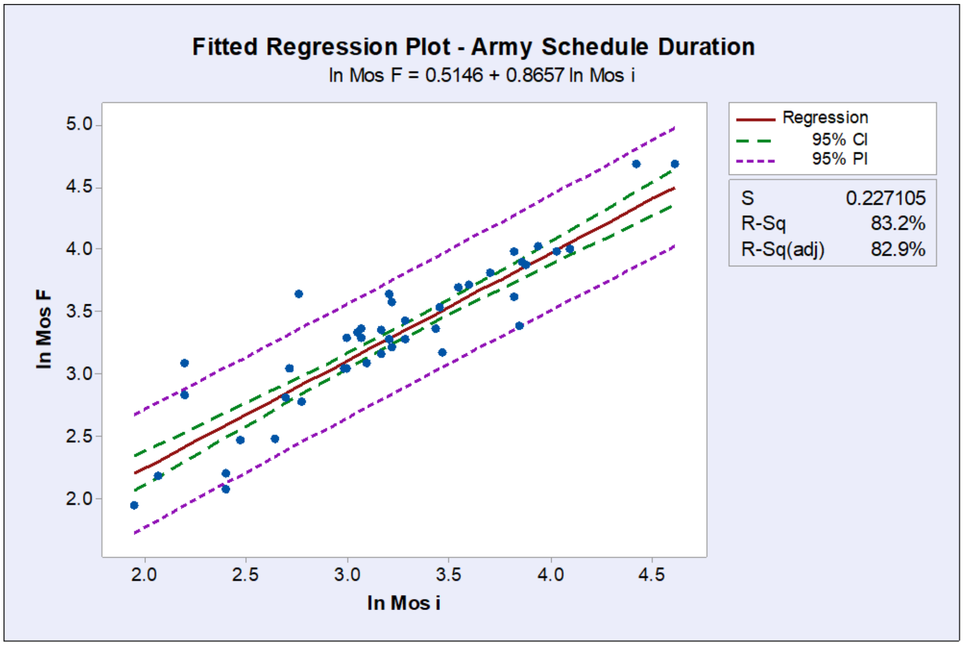


Figure 61: Fitted Regression Plot - Actual Duration (Army)

Results for: AF Schedule Duration (Subset)

Regression Analysis: ln Months_F versus ln Months_i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	3.70572	61.30%	3.70572	3.70572	58.62	0.000
ln Mos i	1	3.70572	61.30%	3.70572	3.70572	58.62	0.000
Error	37	2.33915	38.70%	2.33915	0.06322		
Lack-of-Fit	28	2.32441	38.45%	2.32441	0.08301	50.67	0.000
Pure Error	9	0.01474	0.24%	0.01474	0.00164		
Total	38	6.04487	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.251437	61.30%	60.26%	2.65284	56.11%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	1.085	0.327	(0.421, 1.748)	3.31	0.002	
ln Mos i	0.7258	0.0948	(0.5337, 0.9179)	7.66	0.000	1.00

Regression Equation

```
ln Mos F = 1.0847 + 0.7258 ln Mos i
```

which translates to:

Air Force: $Actual\ Total\ Duration = 2.9587 * (Estimated\ Total\ Duration)^{.7258}$

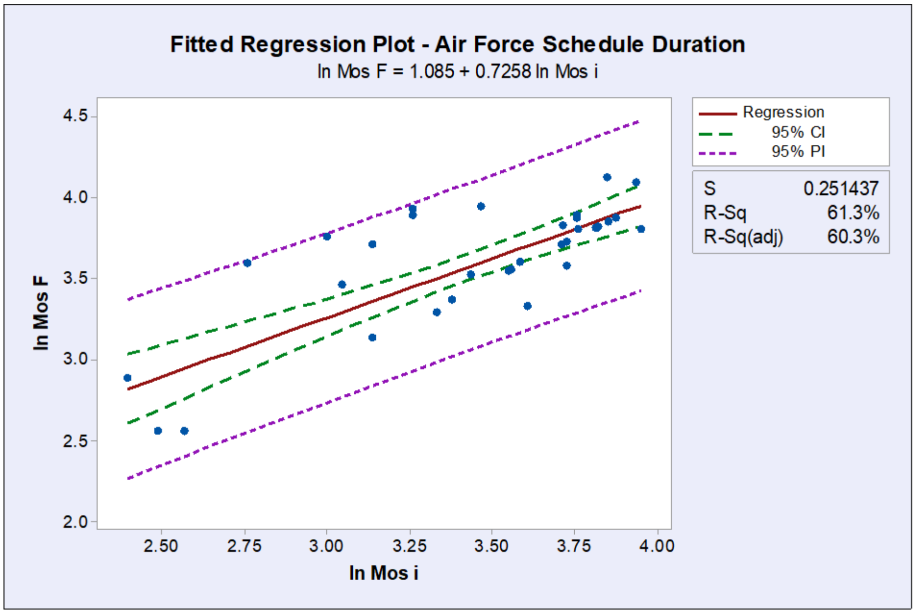


Figure 62: Fitted Regression Plot - Actual Duration (Air Force)

Results for: NAVY Schedule Duration (Subset)

Regression Analysis: ln Months_F versus ln Months_i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	15.5168	79.66%	15.5168	15.5168	215.45	0.000
ln Mos i	1	15.5168	79.66%	15.5168	15.5168	215.45	0.000
Error	55	3.9611	20.34%	3.9611	0.0720		
Lack-of-Fit	42	3.6669	18.83%	3.6669	0.0873	3.86	0.006
Pure Error	13	0.2942	1.51%	0.2942	0.0226		
Total	56	19.4779	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.268367	79.66%	79.29%	4.42219	77.30%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	1.036	0.166	(0.704, 1.368)	6.25	0.000	
ln Mos i	0.7410	0.0505	(0.6399, 0.8422)	14.68	0.000	1.00

Regression Equation

ln Mos F = 1.0361 + 0.7410 ln Mos i

which translates to:

NAVY: $Actual\ Total\ Duration = 2.8182 * (Estimated\ Total\ Duration)^{.7410}$

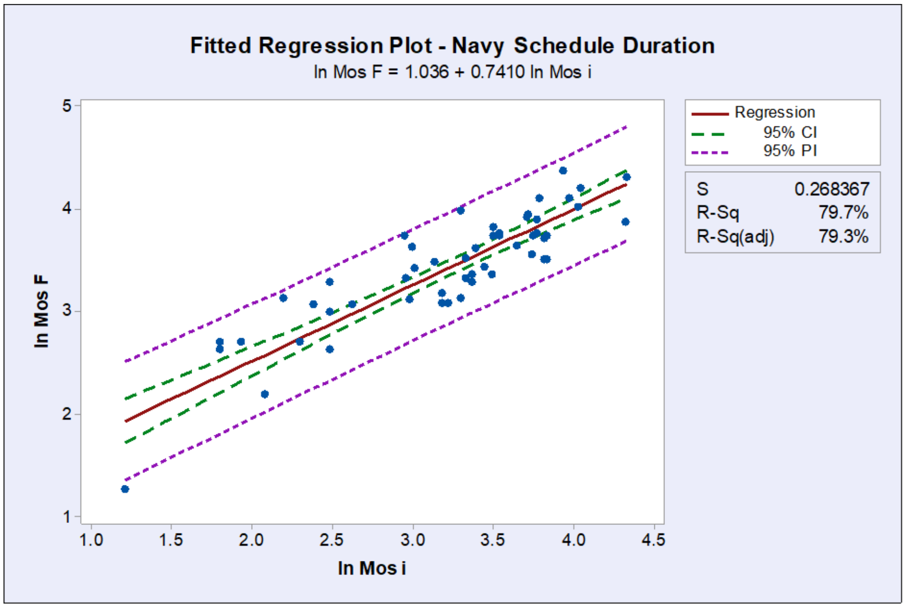


Figure 63: Fitted Regression Plot - Actual Duration (Navy)

Total Hours

Regression Analysis: ln Total Hrs_F versus ln Total Hrs_i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	269.536	89.86%	269.536	269.536	1417.46	0.000
ln Total Hrs_i	1	269.536	89.86%	269.536	269.536	1417.46	0.000
Error	160	30.425	10.14%	30.425	0.190		
Lack-of-Fit	159	30.082	10.03%	30.082	0.189	0.55	0.820
Pure Error	1	0.342	0.11%	0.342	0.342		
Total	161	299.960	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.436066	89.86%	89.79%	31.3742	89.54%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	1.198	0.245	(0.714, 1.682)	4.89	0.000	
ln Total Hrs_i	0.9097	0.0242	(0.8620, 0.9574)	37.65	0.000	1.00

Regression Equation

ln Total Hrs_F = 1.1978 + 0.9097 ln Total Hrs_i

which translates to: $Actual\ Total\ Hours = 3.3128 * (Estimated\ Total\ Hours)^{.9097}$

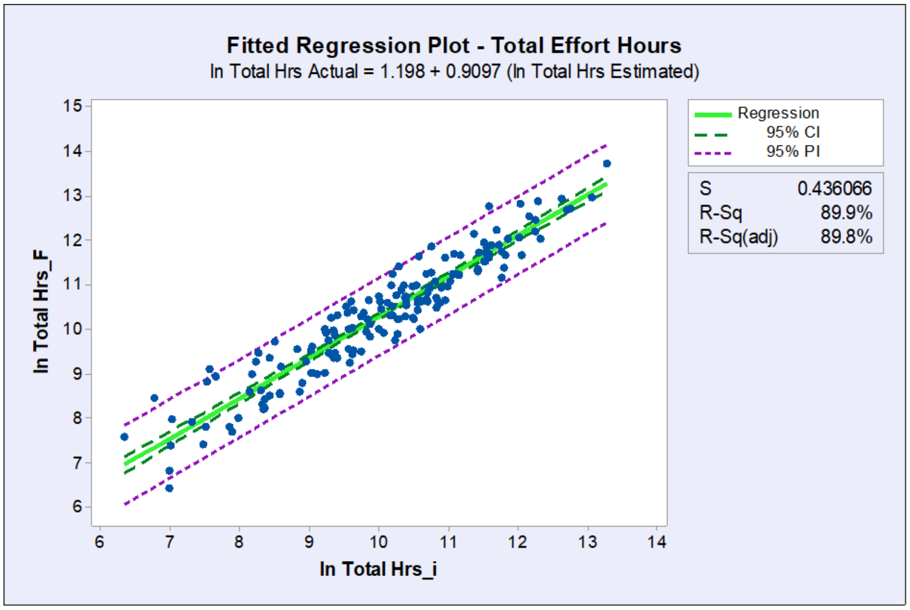


Figure 64: Fitted Regression Plot - Actual Total Effort

The following model is included in case an initial estimate for total hours is not available, but there is an initial estimate of size (ESLOC). The strength of the fit is only moderate.

Regression Analysis: ln Total Hrs_F versus ln ESLOC_i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	202.26	67.43%	202.26	202.265	331.26	0.000
ln ESLOC_i	1	202.26	67.43%	202.26	202.265	331.26	0.000
Error	160	97.70	32.57%	97.70	0.611		
Total	161	299.96	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.781407	67.43%	67.23%	100.336	66.55%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	2.031	0.460	(1.122, 2.940)	4.41	0.000	
ln ESLOC_i	0.8259	0.0454	(0.7363, 0.9155)	18.20	0.000	1.00

Regression Equation:

```
ln Total Hrs_F = 2.0307 + 0.8259 ln ESLOC_i
```

which translates to: $Actual\ Total\ Hours = 7.6192 * (Estimated\ ESLOC)^{.8259}$

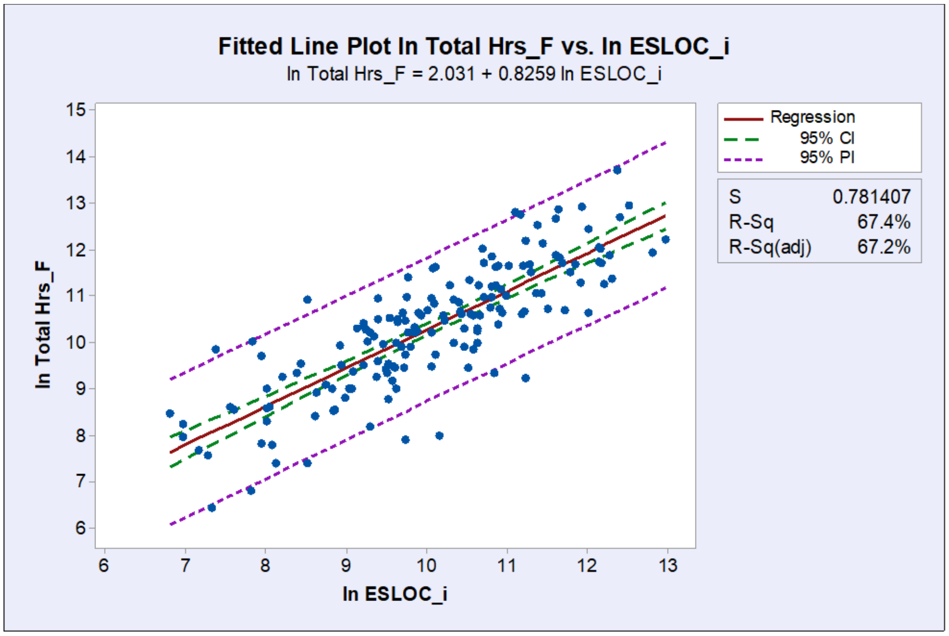


Figure 65: Fitted Regression Plot - Actual Total Effort by ESLOC

Productivity

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	51.25	55.24%	51.25	51.2468	197.49	0.000
ln Prod i	1	51.25	55.24%	51.25	51.2468	197.49	0.000
Error	160	41.52	44.76%	41.52	0.2595		
Total	161	92.76	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.509396	55.24%	54.96%	42.4649	54.22%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	1.221	0.269	(0.689, 1.753)	4.54	0.000	
ln Prod i	0.7439	0.0529	(0.6393, 0.8484)	14.05	0.000	1.00

Regression Equation

```
ln Prod F = 1.2212 + 0.7439 ln Prod i
```

which translates to: $Actual\ Productivity = 3.3914(Estimated\ Productivity)^{.7439}$

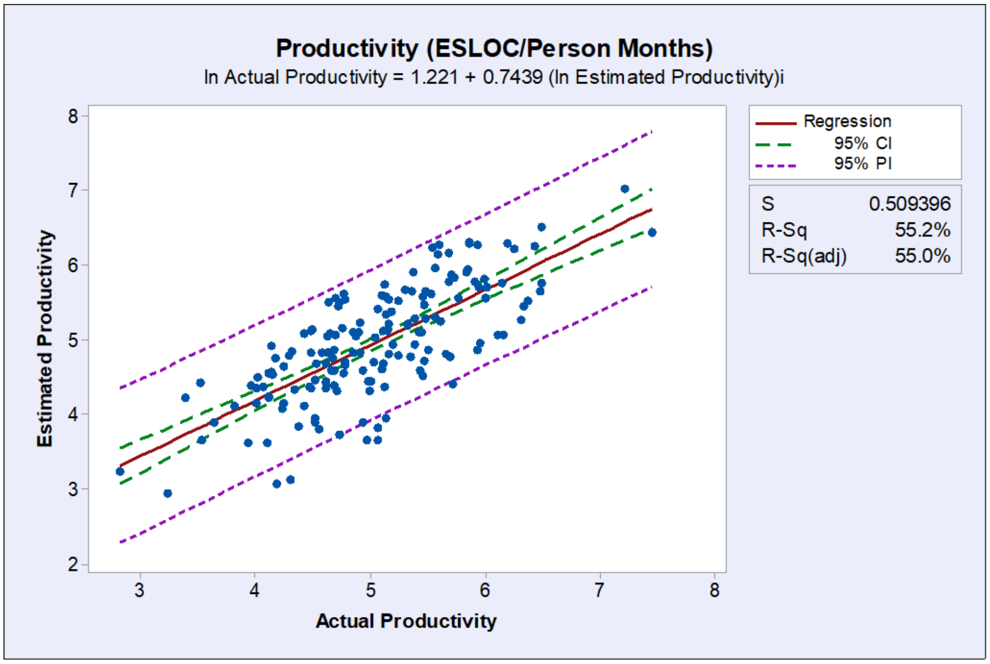


Figure 66: Fitted Regression Plot - Actual Productivity

Results for AIS Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	2.385	49.68%	2.385	2.3853	18.76	0.000
ln Prod i	1	2.385	49.68%	2.385	2.3853	18.76	0.000
Error	19	2.416	50.32%	2.416	0.1272		
Total	20	4.802	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.356614	49.68%	47.03%	2.90187	39.56%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	2.054	0.836	(0.304, 3.804)	2.46	0.024	
ln Prod i	0.665	0.154	(0.344, 0.987)	4.33	0.000	1.00

Regression Equation

ln Prod F = 2.0539 + 0.6651 ln Prod i

which translates to:

AIS: $Actual\ Productivity = 7.7983 * (Estimated\ Productivity)^{.6651}$

Results for ENG Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	8.553	59.02%	8.553	8.5530	25.92	0.000
ln Prod i	1	8.553	59.02%	8.553	8.5530	25.92	0.000
Error	18	5.940	40.98%	5.940	0.3300		
Total	19	14.493	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.574435	59.02%	56.74%	7.09712	51.03%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	1.550	0.693	(0.093, 3.007)	2.24	0.038	
ln Prod i	0.664	0.130	(0.390, 0.938)	5.09	0.000	1.00

Regression Equation

ln Prod F = 1.5502 + 0.6639 ln Prod i

which translates to:

ENG: $Actual\ Productivity = 4.7124(Estimated\ Productivity)^{.6639}$

<u>Results for RT Productivity (Subset)</u>

RT Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	29.48	50.91%	29.48	29.4847	120.29	0.000
ln Prod i	1	29.48	50.91%	29.48	29.4847	120.29	0.000
Error	116	28.43	49.09%	28.43	0.2451		
Total	117	57.92	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.495099	50.91%	50.48%	29.3631	49.30%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	1.360	0.318	(0.730, 1.990)	4.28	0.000	
ln Prod i	0.7027	0.0641	(0.5758, 0.8296)	10.97	0.000	1.00

Regression Equation

ln Prod F = 1.360 + 0.7027 ln Prod i

which translates to:

RT: $Actual\ Productivity = 3.8969(Estimated\ Productivity)^{.7027}$

The following models are based on the difference between the initial estimated productivity and the final actual productivity. As such, it requires data from both the 2630-2 and the 2630-3. It cannot be calculated using only the initial estimate. However, if data can be accessed at some midway point in the development lifecycle, this approach should result in more reliable estimates of the outcome. These models can also be used when considering analogies for estimation by comparing other project attributes.

First we present the model for all cases with a positive change (underestimate) in productivity, followed by the breakout models for AIS, ENG, and RT. Next, we show the model for all cases with a negative change (overestimate) in productivity followed by the super domain breakout models.

Results for Cases with Positive Change in Productivity

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	32.958	88.63%	32.958	32.9578	600.13	0.000
ln Prod i	1	32.958	88.63%	32.958	32.9578	600.13	0.000
Error	77	4.229	11.37%	4.229	0.0549		
Total	78	37.186	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.234345	88.63%	88.48%	4.44599	88.04%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.804	0.181	(0.444, 1.165)	4.44	0.000	
ln Prod i	0.9120	0.0372	(0.8378, 0.9861)	24.50	0.000	1.00

Regression Equation

ln Prod F = 0.8043 + 0.9120 ln Prod i

which translates to: $Actual\ Productivity = 2.235(Estimated\ Productivity)^{.912}\epsilon$

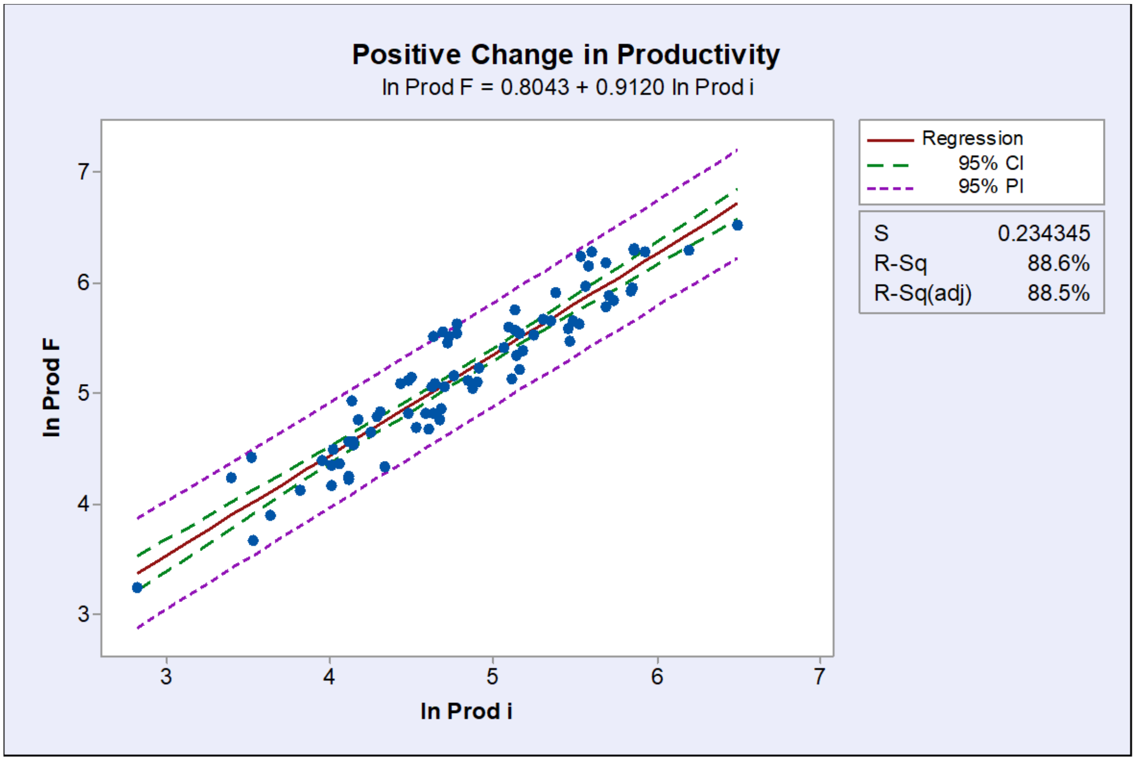


Figure 67: Fitted Regression Plot - Actual Positive Productivity (underestimated)

Results for AIS Cases with Positive Change in Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

n = 13

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	1.6636	76.09%	1.6636	1.66363	35.00	0.000
ln Prod i	1	1.6636	76.09%	1.6636	1.66363	35.00	0.000
Error	11	0.5229	23.91%	0.5229	0.04754		
Total	12	2.1865	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.218025	76.09%	73.91%	0.659752	69.83%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	2.099	0.633	(0.707, 3.491)	3.32	0.007	
ln Prod i	0.698	0.118	(0.439, 0.958)	5.92	0.000	1.00

Regression Equation

ln Prod F = 2.0991 + 0.6983 ln Prod i

which translates to:

AIS: $Actual\ Productivity = 8.1589 * (Estimated\ Productivity)^{..6983}$

Results for ENG Cases with Positive Change in Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

n = 9

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	3.0973	92.38%	3.0973	3.09729	84.83	0.000
ln Prod i	1	3.0973	92.38%	3.0973	3.09729	84.83	0.000
Error	7	0.2556	7.62%	0.2556	0.03651		
Total	8	3.3529	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.191079	92.38%	91.29%	0.467369	86.06%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.030	0.542	(-1.251, 1.312)	0.06	0.957	
ln Prod i	1.085	0.118	(0.806, 1.363)	9.21	0.000	1.00

Regression Equation

ln Prod F = 0.0302 + 1.0848 ln Prod i

which translates to:

ENG: $Actual\ Productivity = 1.0307 * (Estimated\ Productivity)^{1.0848}$

Results for RT Cases with Positive Change in Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

n = 55

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	20.392	88.01%	20.392	20.3920	389.13	0.000
ln Prod i	1	20.392	88.01%	20.392	20.3920	389.13	0.000
Error	53	2.777	11.99%	2.777	0.0524		
Total	54	23.169	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.228920	88.01%	87.79%	2.98327	87.12%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF

Constant	0.885	0.213	(0.457, 1.313)	4.15	0.000	
ln Prod i	0.8873	0.0450	(0.7971, 0.9775)	19.73	0.000	1.00

Regression Equation

ln Prod F = 0.885 + 0.8873 ln Prod i

which translates to:

RT: $Actual\ Productivity = 2.4233 * (Estimated\ Productivity)^{.8873}$

Results for Cases with Negative Change in Productivity

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	36.09	75.79%	36.09	36.0862	253.56	0.000
ln Prod i	1	36.09	75.79%	36.09	36.0862	253.56	0.000
Error	81	11.53	24.21%	11.53	0.1423		
Total	82	47.61	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.377252	75.79%	75.49%	12.0880	74.61%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.077	0.296	(-0.512, 0.666)	0.26	0.795	
ln Prod i	0.8910	0.0560	(0.7797, 1.0023)	15.92	0.000	1.00

Regression Equation

ln Prod F = 0.0771 + 0.8910 ln Prod i

which translates to:

$$Actual\ Productivity = 1.0802 * (Estimated\ Productivity)^{.891}$$

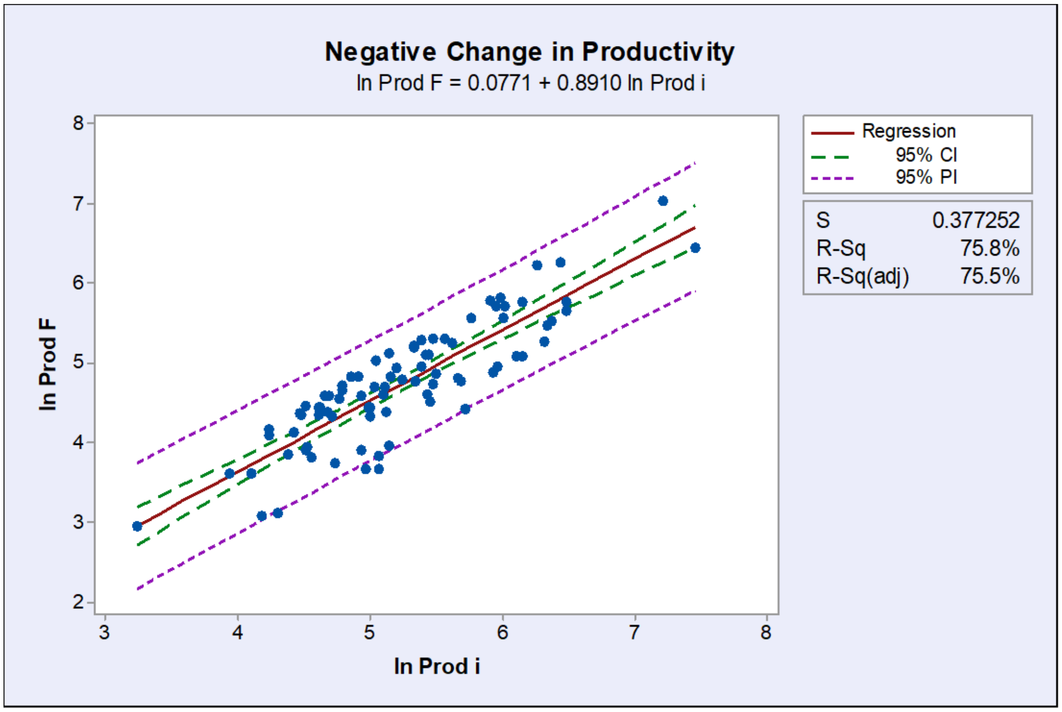


Figure 68: Fitted Regression Plot - Actual Negative Productivity (overestimated)

Results for AIS Cases with Negative Change in Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	1.66477	98.46%	1.66477	1.66477	383.74	0.000
ln Prod i	1	1.66477	98.46%	1.66477	1.66477	383.74	0.000
Error	6	0.02603	1.54%	0.02603	0.00434		
Total	7	1.69080	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.0658660	98.46%	98.20%	0.0452579	97.32%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	-0.078	0.280	(-0.763, 0.608)	-0.28	0.791	
ln Prod i	0.9832	0.0502	(0.8604, 1.1061)	19.59	0.000	1.00

Regression Equation

```
ln Prod F = -0.078 + 0.9832 ln Prod i
```

which translates to:

$$AIS\ Actual\ Productivity = 0.9254 * (Estimated\ Productivity)^{.9832}$$

Results for ENG Cases with Negative Change in Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	9.640	86.67%	9.640	9.6399	58.52	0.000
ln Prod i	1	9.640	86.67%	9.640	9.6399	58.52	0.000
Error	9	1.483	13.33%	1.483	0.1647		
Total	10	11.123	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.405875	86.67%	85.19%	2.35519	78.83%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	-0.693	0.760	(-2.412, 1.027)	-0.91	0.386	
ln Prod i	0.996	0.130	(0.701, 1.290)	7.65	0.000	1.00

Regression Equation

ln Prod F = -0.693 + 0.9958 ln Prod i

which translates to:

$$ENG\ Actual\ Productivity = 0.5001 * (Estimated\ Productivity)^{.9958}$$

Results for RT Cases with Negative Change in Productivity (Subset)

Regression Analysis: ln Prod F versus ln Prod i

Analysis of Variance

Source	DF	Seq SS	Contribution	Adj SS	Adj MS	F-Value	P-Value
Regression	1	20.512	70.84%	20.512	20.5125	148.18	0.000
ln Prod i	1	20.512	70.84%	20.512	20.5125	148.18	0.000
Error	61	8.444	29.16%	8.444	0.1384		
Total	62	28.957	100.00%				

Model Summary

S	R-sq	R-sq(adj)	PRESS	R-sq(pred)
0.372058	70.84%	70.36%	8.97412	69.01%

Coefficients

Term	Coef	SE Coef	95% CI	T-Value	P-Value	VIF
Constant	0.302	0.357	(-0.411, 1.015)	0.85	0.400	
ln Prod i	0.8431	0.0693	(0.7046, 0.9816)	12.17	0.000	1.00

Regression Equation

ln Prod F = 0.302 + 0.8431 ln Prod i

which translates to:

$$RT\ Actual\ Productivity = 1.3529 * (Estimated\ Productivity)^{.8431}$$

Appendix G: Burden Labor Rate

A burden labor rate is used in this analysis to derive cost. The rate includes:

- wages
- payroll taxes
- worker's compensation and health insurance
- paid time off
- training and travel expenses
- vacation and sick leave
- pension contributions
- and other benefits

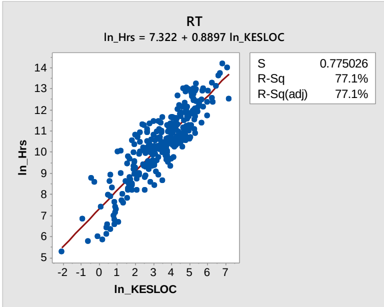
The burdened rate may be as much as 50% higher than payroll costs alone (i.e., more than 50% of wages).

An average burden labor rate of $150,000 per year is assumed. This rate breaks down to $12,500/month and $82.24/hour using 1,824 labor hours in a year. The 1,824 labors hours is based on 152 labor hours per month for 12 months.

Appendix H: Most-Least Expensive Software Analysis Details

Average project size for this data set is 40,000 ESLOC or 40 KESLOC. The natural log equivalent is 10.6 ln_ESLOC or 3.69 ln_KESLOC, respectively.

RT Regression Analysis: ln_Hrs versus ln_KESLOC



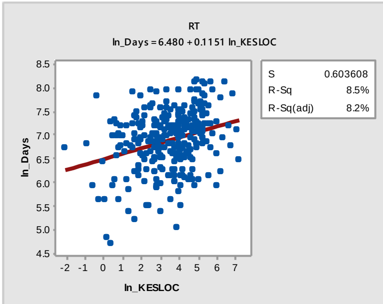
The regression equation is
ln_Hrs = 7.322 + 0.8897 ln_KESLOC

S = 0.775026   R-Sq = 77.1%   R-Sq(adj) = 77.1%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	577.565	577.565	961.54	0.000
Error	285	171.190	0.601		
Total	286	748.754			

RT Regression Analysis: ln_Days versus ln_KESLOC



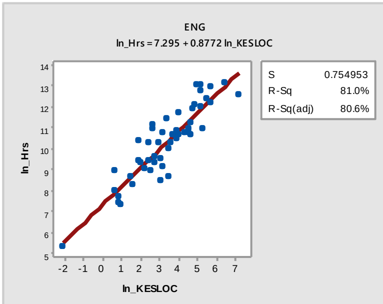
The regression equation is
ln_Days = 6.480 + 0.1151 ln_KESLOC

S = 0.603608   R-Sq = 8.5%   R-Sq(adj) = 8.2%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	9.663	9.66275	26.52	0.000
Error	285	103.838	0.36434		
Total	286	113.500			

ENG Regression Analysis: ln_Hrs versus ln_KESLOC



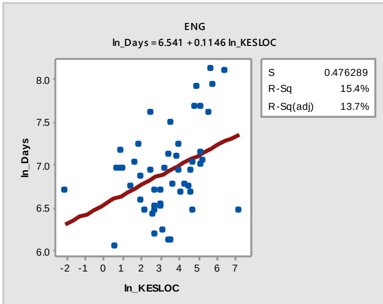
The regression equation is
ln_Hrs = 7.295 + 0.8772 ln_KESLOC

S = 0.754953   R-Sq = 81.0%   R-Sq(adj) = 80.6%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	116.460	116.460	204.33	0.000
Error	48	27.358	0.570		
Total	49	143.818			

ENG Regression Analysis: ln_Days versus ln_KESLOC



The regression equation is
ln_Days = 6.541 + 0.1146 ln_KESLOC

S = 0.476289   R-Sq = 15.4%   R-Sq(adj) = 13.7%

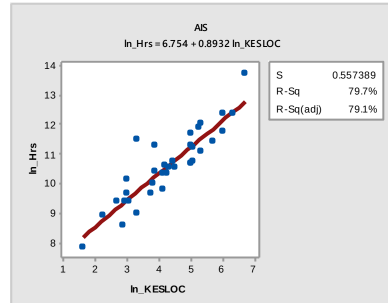
Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	1.9884	1.98839	8.77	0.005
Error	48	10.8889	0.22685		
Total	49	12.8773			

AIS Regression Analysis: ln_Hrs versus ln_KESLOC

The regression equation is
ln_Hrs = 6.754 + 0.8932 ln_KESLOC

S = 0.557389   R-Sq = 79.7%   R-Sq(adj) = 79.1%



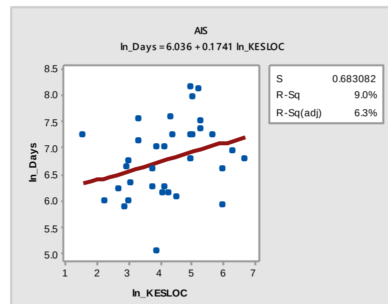
Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	40.1725	40.1725	129.30	0.000
Error	33	10.2525	0.3107		
Total	34	50.4250			

AIS Regression Analysis: ln_Days versus ln_KESLOC

The regression equation is
ln_Days = 6.036 + 0.1741 ln_KESLOC

S = 0.683082   R-Sq = 9.0%   R-Sq(adj) = 6.3%



Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	1.5267	1.52668	3.27	0.080
Error	33	15.3979	0.46660		
Total	34	16.9245			

Appendix I: Best-in-class/Worst-in-class Software Analysis Details

Average project size for this data set differed by super-domain. Table 32 shows the average sizes and their natural log equivalents.

Table 32: Super-Domain Average project Size

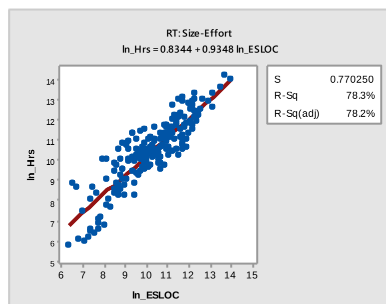
	Average Size	Ln Equivalent
Real Time (RT)	34,000	10.43
Engineering (ENG)	32,000	10.37
Automated Information Systems (AIS)	72,000	11.18

RT Average Project Size

```
Variable     N  N*     Mean  SE Mean  StDev  Minimum     Q1  Median      Q3  Maximum
ln_ESLOC   198   0   10.289    0.111  1.560    6.317  9.175  10.445  11.459   14.047
Mean: 29,407
Median: 34,372
```

Average project size is: 34,000 ESLOC

RT Regression Analysis: ln_Hrs versus ln_ESLOC



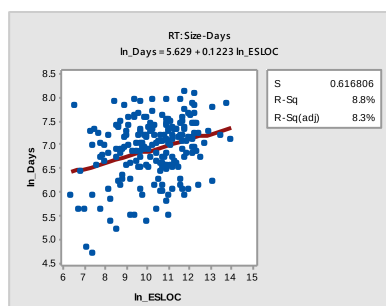
The regression equation is
ln_Hrs = 0.8344 + 0.9348 ln_ESLOC

S = 0.770250 R-Sq = 78.3% R-Sq(adj) = 78.2%

Analysis of Variance

```
Source      DF      SS       MS       F      P
Regression   1  419.108  419.108  706.42  0.000
Error      196  116.284    0.593
Total      197  535.392
```

RT Regression Analysis: ln_Days versus ln_ESLOC



The regression equation is
ln_Days = 5.629 + 0.1223 ln_ESLOC

S = 0.616806 R-Sq = 8.8% R-Sq(adj) = 8.3%

Analysis of Variance

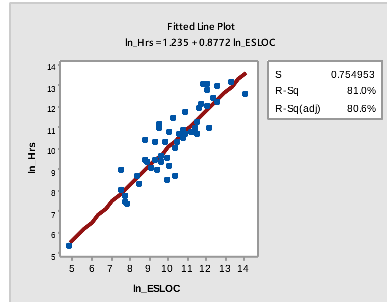
```
Source       DF       SS       MS      F      P
Regression    1   7.1682  7.16816  18.84  0.000
Error       196  74.5682  0.38045
Total       197  81.7363
```

ENG Average Project Size

```
Variable    N  N*     Mean  SE Mean  StDev  Minimum     Q1  Median      Q3  Maximum
ln_ESLOC   50   0   10.314    0.249  1.757    4.851  9.319  10.388  11.632   14.099
Mean: 30,152
Median: 32,468
```

The average project size is: 32,000

ENG Regression Analysis: ln_Hrs versus ln_ESLOC



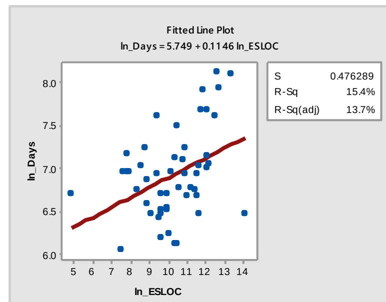
The regression equation is
ln_Hrs = 1.235 + 0.8772 ln_ESLOC

S = 0.754953   R-Sq = 81.0%   R-Sq(adj) = 80.6%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	116.460	116.460	204.33	0.000
Error	48	27.358	0.570		
Total	49	143.818			

ENG Regression Analysis: ln_Days versus ln_ESLOC



The regression equation is
ln_Days = 5.749 + 0.1146 ln_ESLOC

S = 0.476289   R-Sq = 15.4%   R-Sq(adj) = 13.7%

Analysis of Variance

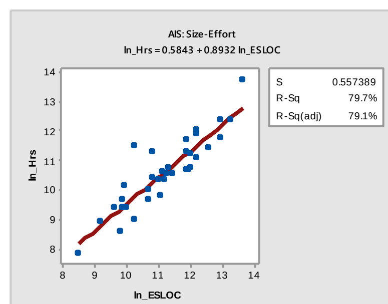
Source	DF	SS	MS	F	P
Regression	1	1.9884	1.98839	8.77	0.005
Error	48	10.8889	0.22685		
Total	49	12.8773			

AIS Average Project Size

Variable	N	N*	Mean	SE Mean	StDev	Minimum	Q1	Median	Q3	Maximum
ln_ESLOC	35	0	**11.198**	0.206	1.217	8.475	10.215	**11.180**	11.994	13.641

Mean = 72,984
Median = 71,682
Average project size = 72,000

AIS Regression Analysis: ln_Hrs versus ln_ESLOC



The regression equation is

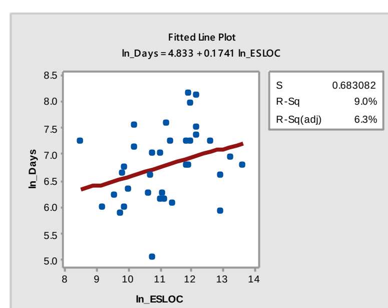
ln_Hrs = 0.5843 + 0.8932 ln_ESLOC

S = 0.557389   R-Sq = 79.7%   R-Sq(adj) = 79.1%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	40.1725	40.1725	129.30	0.000
Error	33	10.2525	0.3107		
Total	34	50.4250			

AIS Regression Analysis: ln_Days versus ln_ESLOC



The regression equation is
ln_Days = 4.833 + 0.1741 ln_ESLOC

S = 0.683082   R-Sq = 9.0%   R-Sq(adj) = 6.3%

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	1.5267	1.52668	3.27	0.080
Error	33	15.3979	0.46660		
Total	34	16.9245			

Appendix J: Initial/Final Cases with Complete Schedule Change Data

Table 33 shows the changes in schedule for those cases with complete schedule data reported by phase.

Table 33: Summary of Schedule Change (in Months) – SRDR Pairs with Complete Phase Data

ALL cases (58)	**Change in Schedule (Actual - Estimate) by SW Development Phase (in months)**						
Change in	**Mean**	**Median**	**Minimum value**	**Maximum value**	**no change reported**	***n* reporting change**	**missing**
Start Date	0	0	-2	12	46	12	0
End Date	3	0	-43	57	19	39	0
Total Duration	14	8	-42	78	8	58	0
Total Hours	12,998	1,201	-56,778	350,591	0	58	0
Reqs End Date	8	2	-45	58	10	48	0
Arch End Date	5	3	-10	58	14	44	0
Code End Date	12	9	-9	61	4	54	0
INT End Date	5	1	-17	59	12	46	0
Qual End Date	4	1	-42	59	10	48	0
DTE End Date	4	1	-20	56	10	48	0

Table 34 shows an increasing correlation in schedule change (i.e., when schedule changes in one phase, succeeding phases also experience a schedule change).[19] The correlation of change generally *increases* with each succeeding software development phase (requirements to architecture to coding to integration to qualification testing and to development test and evaluation (DTE).

Table 34: Change in Schedule Correlations

	End Date	Total Hours	Total Duration	Reqs End Date	Arch End Date	Code End Date	INT End Date	Qual End Date
Total Hours	0.368							
p-value	*0.005*							
Total Duration	0.517	0.082						
p-value	*0*	*0.543*						
Reqs End Date	0.489	0.358	0.159					
p-value	*0*	*0.006*	*0.233*					
Arch End Date	**0.618**	0.559	0.243	**0.686**				
p-value	*0*	*0*	*0.066*	*0*				
Code End Date	**0.607**	0.439	0.233	0.513	**0.752**			
p-value	*0*	*0.001*	*0.079*	*0*	*0*			
INT End Date	**0.74**	0.48	0.36	0.417	0.816	**0.794**		
p-value	*0*	*0*	*0.006*	*0.001*	*0*	*0*		
Qual End Date	**0.908**	0.422	0.5	0.545	0.694	0.711	**0.84**	
p-value	*0*	*0.001*	*0*	*0*	*0*	*0*	*0*	
DTE End Date	**0.848**	0.392	0.402	0.362	0.727	0.643	0.865	**0.787**
p-value	*0*	*0.002*	*0.002*	*0.005*	*0*	*0*	*0*	*0*

19 The data used for this analysis was limited due to the constraints that an initial and final SRDR pair had to exist and there had to be values for each software development phase in the initial and final data. This resulted in 58 pairs of data being analyzed.

The conclusion is that once schedule begins to slip, the slip propagates to succeeding phases.

The analysis of growth revealed several things:

- Additional requirements is associated with an increase in productivity.
- The largest size increases occurred in projects that experienced a positive productivity increase between initial and final SRDRs.
- Projects with positive productivity showed the strongest median value increase in duration of 42%.
- Negative productivity group projects showed the most increase in expended hours between initial and final SRDRs, 54%.
- Once schedule begins to slip, succeeding phases also slip in schedule.

During the course of the analyses, several results suggested further study and analyses. There is more to understand in comparing initial SRDR data to final SRDR data which is beyond the scope of this current effort. Future research should investigate:

- An area of special analytical interest is the difference between those projects/builds which performed better than expected versus those that did not. One measure which drives cost is productivity. Estimates of productivity which are higher than achieved can drive cost/schedule overruns. The following tables give breakdowns of several of the key variables for all pairs followed by the same breakdowns by those cases which under-performed their productivity estimates and the cases which performed better than their productivity estimates.
- Another area of analytical interest is the cascading effect schedule slippage as seen in Table 34. There is very limited data with complete information for all software development phases. Interestingly, changes increase as development phases progress. However, while this effect is mirrored with the change in end date we do not see this relationship to total hours. Future effort could further investigate this cascading effect by expanding the dataset with closer examination of cases with incomplete data.

Appendix K: Data Source Details

Background

A spreadsheet of transcribed DACIMS SRDR data that was produced by the Naval Air Systems Command (NAVAIR) and dated July 2013 forms the basis of analysis in this document. Under the leadership of Michael Popp, NAVAIR evaluated the contractor submissions regularly to incorporate new data, which they used to establish/revise cost estimating relationships. This team's activity provided a valuable service to the Department of Defense (DoD) cost community. Efforts are underway to replicate this activity at the Service cost centers. Wilson Rosa (formerly with the Air Force Cost Analysis Agency, now with the Naval Center for Cost Analysis) used a version of this spreadsheet to investigate project performance and cost estimating relationships of interest to the Air Force.

The spreadsheet produced by NAVAIR dated July 2013 comprised 2,445 entries transcribed from the original contractor submissions. SEI also obtained a copy of all SRDR files submitted to DCARC as of September 2013. After removing duplicates, these 1679 files include initial and final Software Resource Data Reports (SRDRs), data dictionary files, validation memos from the Defense Cost and Resource Center (DCARC), and other auxiliary information files sometimes provided by the contactors. SEI constructed a repository of the contractor submissions which mirrored the structure of the Defense Automated Cost Information Management System (DACIMS) on the DCARC website as of September 2013 (http://dcarc.cape.osd.mil/csdr/default.aspx).

To facilitate research, SEI cross-linked over 90% of the source documents (contractor submissions) obtained from DCARC to the entries in the NAVAIR spreadsheet and to the Rosa revisions. This enables quick traceability between the source, NAVAIR, and Rosa data whenever issues arise concerning specific entries. SEI also constructed a programming tool for verification purposes. This tool successfully extracts the information from the standard Excel form for 2630-2 (initial) and 2630-3 (final) reports and stores the data in a usable format in a Microsoft Access database, along with the appropriate link addresses. Unfortunately, contractors are able to generate their own variations of the form so that a tool making the data in the files comparable requires a great deal of manual effort. The SEI tool requires further development to address all the different formats and file types, but did extract 1,632 separate entries from 462 files that complied with the standard Excel format. We also cross-linked the appropriate data dictionaries.

The NAVAIR spreadsheet is filled with more than 1,100 comments inserted to help explain and assess particular contractor entries. Much of the data reported is considered suspicious for analytical purposes and NAVAIR has indicated which entries it considers good for use. When at AFCAA, Wilson Rosa further evaluated several of the submissions for "reasonableness," which led him to contact several of the contractor development teams directly for clarification and revision. These communications resulted in several corrections to specific values in the dataset. In our early collaboration with Wilson Rosa, we have incorporated these revisions to the NAVAIR July 2013 dataset. However, the Rosa dataset was based on data collected in 2012. For traceability of the revisions, Wilson Rosa also made available the notes and emails of his communications with the various contractors.

We have identified all the differences between the Rosa dataset and the NAVAIR dataset and have accepted Rosa's revisions when they seemed appropriate. Data evaluated by NAVAIR forms the vast bulk of the data, but for verification, SEI undertook the linkage of actual source documents provided by CAPE/DCARC to the dataset

(CAPE stands for Cost Assessment and Program Evaluation). The dataset we constructed thus uses original contractor submission data as represented in the NAVAIR spreadsheet with specific revisions as communicated to Wilson Rosa by the contractors, along with a few revisions we made based on the original contractor reports.

The selection of which case to use for analysis is the crux of the problem with this type of data. The DACIMS SRDR repository maintained by CAPE/DCARC comprised over 1,700 files in 2013. The data came in various types of files (Excel, Word, PDF, PowerPoint) and in various data formats. Some files included one submission while other files included dozens. The task performed by NAVAIR in transforming this data into a usable form represented considerable effort. Of the 2,445 contractor submissions in the NAVAIR dataset a mix of only 638 initial and final submissions were recommended BY NAVAIR as good for use. We took this as a starting point for the analysis of actual project performance as represented by the contractors' final submissions. Similarly, NAVAIR identified 394 cases as suitable for pairing, that is, the comparison of estimated versus actual performance as represented by the difference between the initial and final submissions.

We performed our own assessment of the data by selecting cases to use for analysis based on the NAVAIR and Rosa recommendations and comments but also using the information contained in the submitted data dictionaries. Our dataset differs slightly from both the NAVAIR and Rosa datasets in this regard since we used our best judgment in comparing all these sources of information for the selection process. Of the 441 final submissions rated good for use by NAVAIR, we created 287 records for analysis. Of the 197 pairs rated good, we selected 181 after our investigation of the data. We maintain a linked database of the NAVAIR and Rosa spreadsheets together with the original source submissions and data dictionaries. All revisions made by any party are identified and traceable to the source documents. The inclusion of all comments by NAVAIR, AFCAA, and SEI should prove useful to any analyst wishing to make use of the data.

Data Demographics

The Software Resources Data Report (SRDR) is the primary source of data on software projects and their performance. It is a contract data deliverable that formalized the reporting of software metrics data. It consists of the following two forms:

1. Data Report
2. Data Dictionary

It is designed to record both the estimates and actual results of new software developments or upgrades.

The SRDR applies to all major contracts and subcontracts, regardless of contract type, for contractors developing or producing software elements within Acquisition Category (ACAT) I and IA programs and pre-MDAP and pre-MAIS programs subsequent to Milestone A approval for any software development element with a projected software effort greater than $20M.[20]

Reporting Frequency

Projects submit reports for two types of reporting events:

20 CSDR Requirements, OSD Defense Cost and Resource Center, http://dcarc.cape.osd.mil/CSDR/CSDROverview.aspx#Introduction

- contract event—an SRDR is required at contract start (Initial Developer Submission, Form 2630-2) and at contract completion (Final Developer Submission)
- product event—an SRDR is required at the start of a product increment (Initial Developer Submission) and at the completion of a product increment (Final Developer Submission, Form 2630-3). An increment is a partial delivery of a product capability. Increments are also referred to as spirals, builds, and releases.

The SRDRs for the start and end of a contract event will contain all of the data for all product events within the contract. Therefore, care must be taken to analyze only records that are from either contract events or product events but not both.

The SRDR event data used in this analysis is based on *product event* data and is referred to as *project* data in this Factbook.

The SRDR data used in this analysis is based on the final report that contains actual result data. Data for this analysis had to include the following information:

- size data
- effort data
- schedule data

Based on this criterion, the dataset for this analysis used 287 projects from the product-event final report data. Similarly, we used 181 pairs of initial and final cases for analysis of the estimated versus actual performance. See Appendix J for details on the paired data.

As more data is added to the Defense Automated Cost Information Management System (DACIMS), this analysis can be expanded and updated.

Distribution by Service

The analysis dataset is spread across the three services (Marine Corps projects are included with Navy projects):

- Army (15)
- Air Force (12)
- Navy (18)

Note, that each program rather than each submission is counted only once.

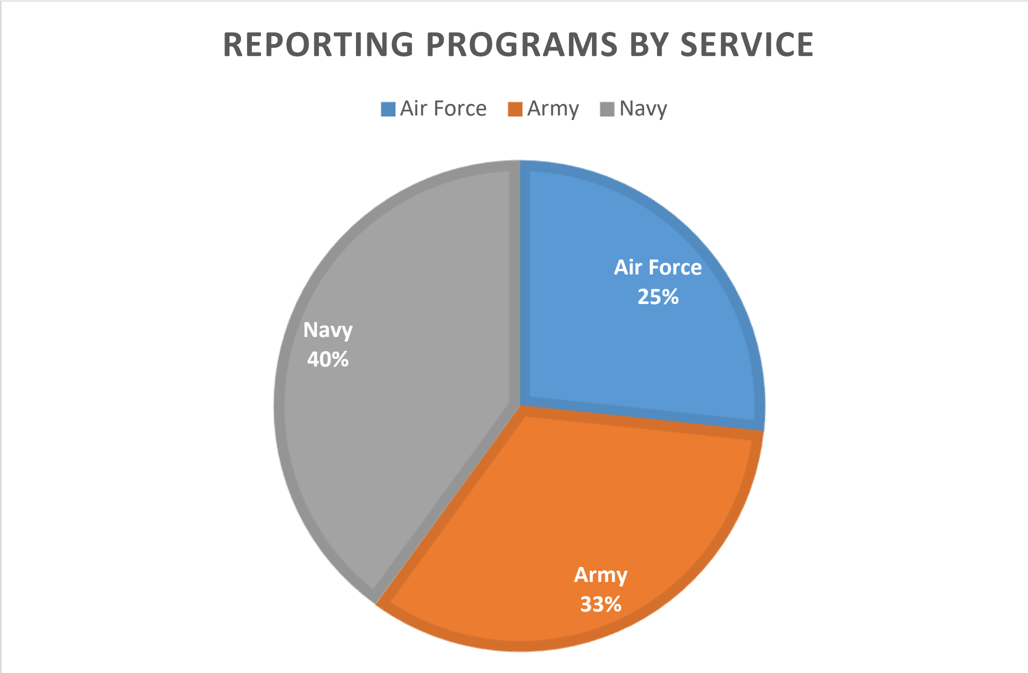


Figure 69: SRDR Final Submissions by Service

1.4 Distribution by Super-domain

The analysis dataset can be segregated into different classes called super-domains. Super-domains are high-level groupings of software application domains, as shown in Figure 70. We initially determined four super-domains:

- engineering software (50)
- real time software (198)
- automated information system software (35)
- mission support software (4)

The Mission Support domain is omitted from the analyses in this report due to its small number of projects. A more detailed explanation of the super-domains is provided in the Appendix C: Super-domains.

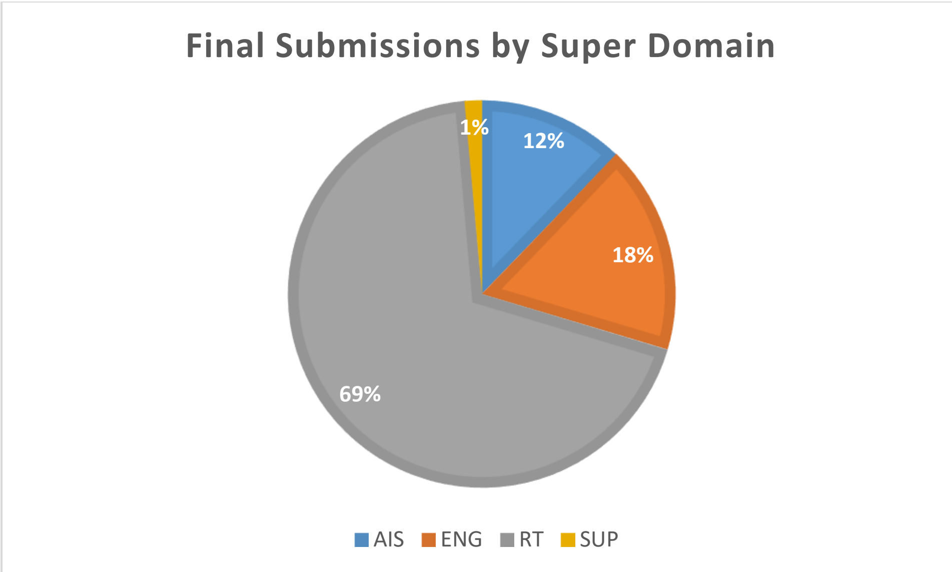


Figure 70: Final Submission by Super Domain

Distribution by Application Domains

Super-domains are a categorization of the thirteen application domains which are identified on the contractor submissions. The following chart lists the application domains and are color coded to indicate the super-domain category. Real Time Embedded, Command and Control, and Signal Processing make up more than half of the entries.

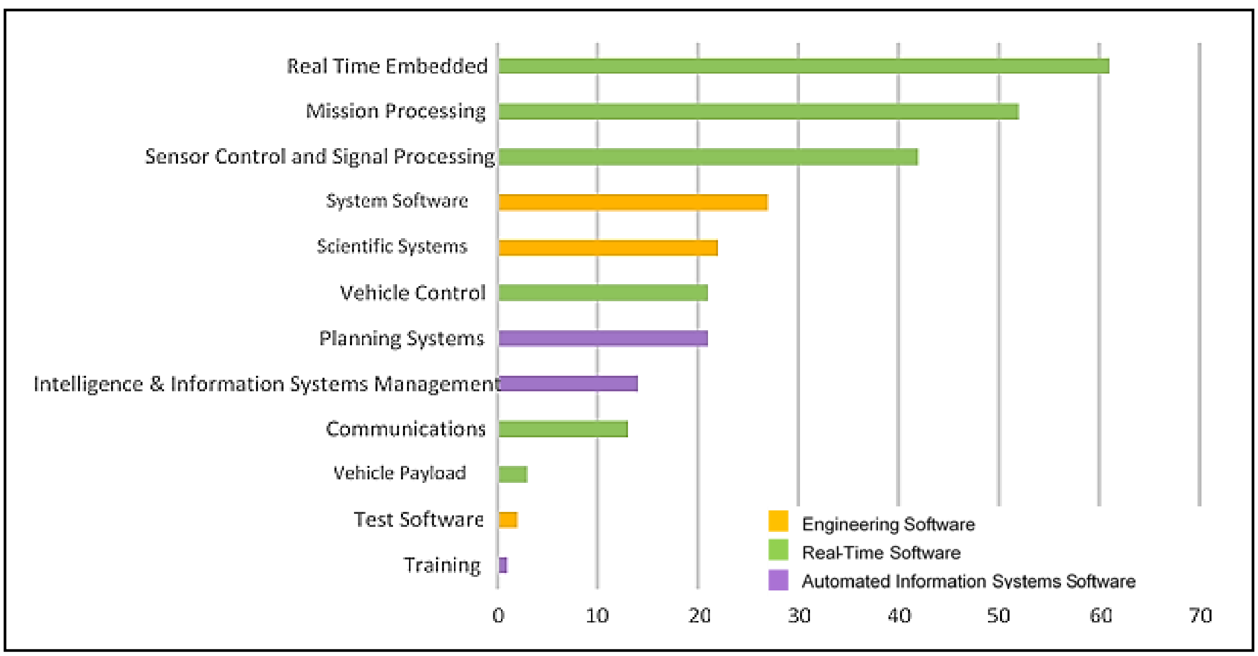


Figure 71: Program Distribution by Application and Super Domain

Given the limited number of data points in some of the domains, the analysis in this report was conducted on Super domains. Overall, the user should consider the results in the Factbook to be most relevant to the individual domains containing the most data points (i.e., the summary data are most likely to resemble Real Time Embedded Projects).

Distribution by Operating Environment

The analysis dataset can also be grouped into the operating environments (OpEnv) in which the software operates, as shown in Figure 4. The most common environment was Mobile followed by Aerial Vehicle.

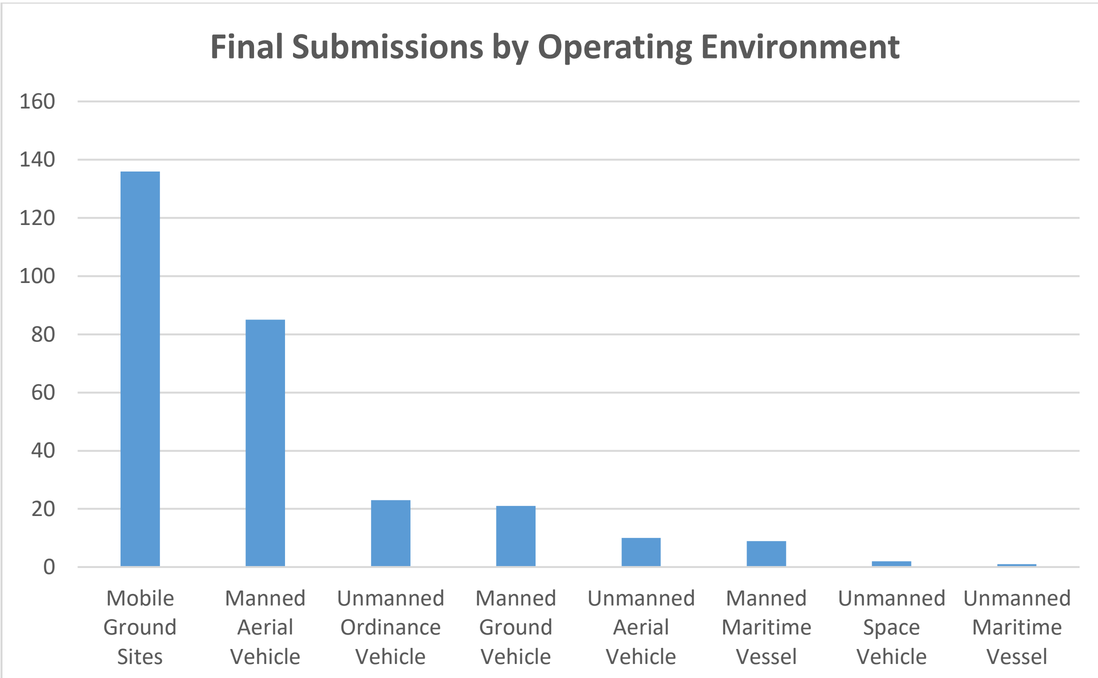


Figure 72: Project Distribution by Operating Environment

Examples of these environments are provided in Appendix D, Operating Environments.

Distribution by Programming Language

Programming languages are shown in the following chart. By far, the C families dominate, which includes C, ANSI C, C++, C#, C/Assembly, and C# Net. Ada still represents a significant portion of software development, which continues to be problematic for future efforts since Ada is no longer commonly taught or supported outside of legacy DoD applications.

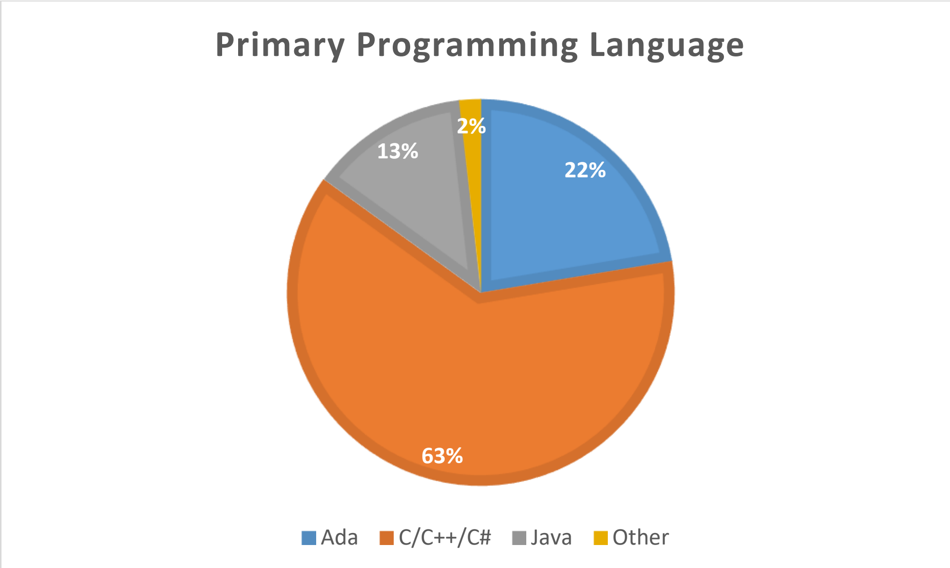


Figure 73: Program Distribution by Language Family

If a Program is using a programming language other than C, Ada, or Java, the analysis in this Factbook will need to be normalized to account for the impact to ESLOC heuristics.

Reported Software Process Maturity Levels

In figure 6, the histogram shows the reported maturity levels in the analysis dataset. Most projects reported the highest level of maturity. The following are the counts at each maturity level:

- Level 2 (3)
- Level 3 (122)
- Level 4 (23)
- Level 5 (221)
- Not Available (37)

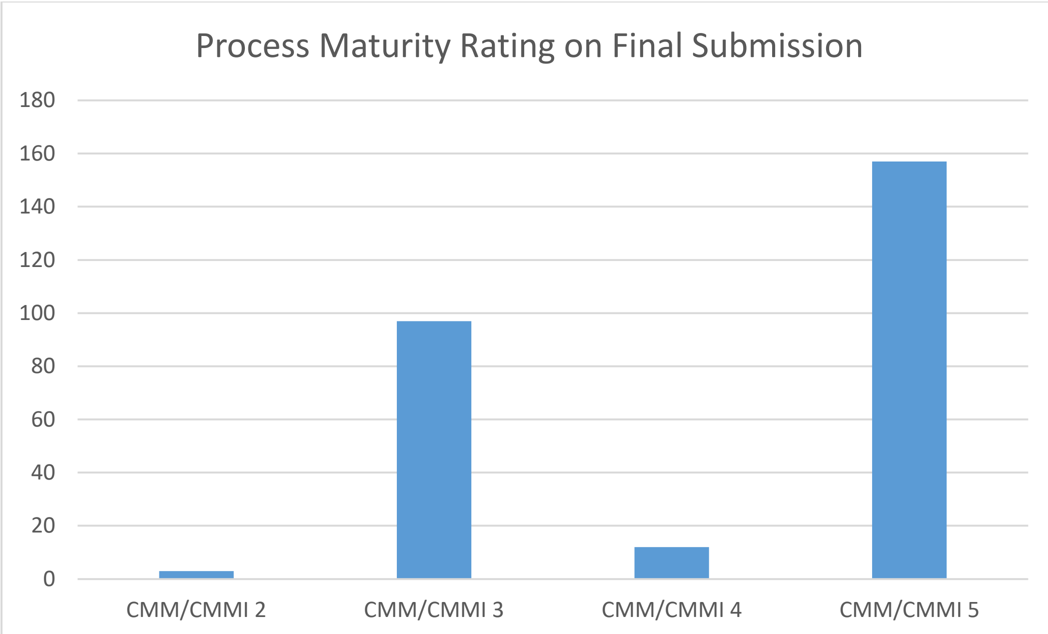


Figure 74: Reported Maturity Levels

Given the majority of the data used to generate the findings in this report comes from higher maturity programs, it would be suspect for a Program to forecast greater performance or productivity than cited in this report by claiming they are operating at a higher maturity level.

Data Age

The age of the data was derived from the Report As Of date. Submission dates in the analysis dataset of the Final Developer Report range from July 2001 to January 2013. As Figure 75 shows, there are a few projects from 2001 - 2004. Most of the projects are from the 2007 to 2012 timeframe.

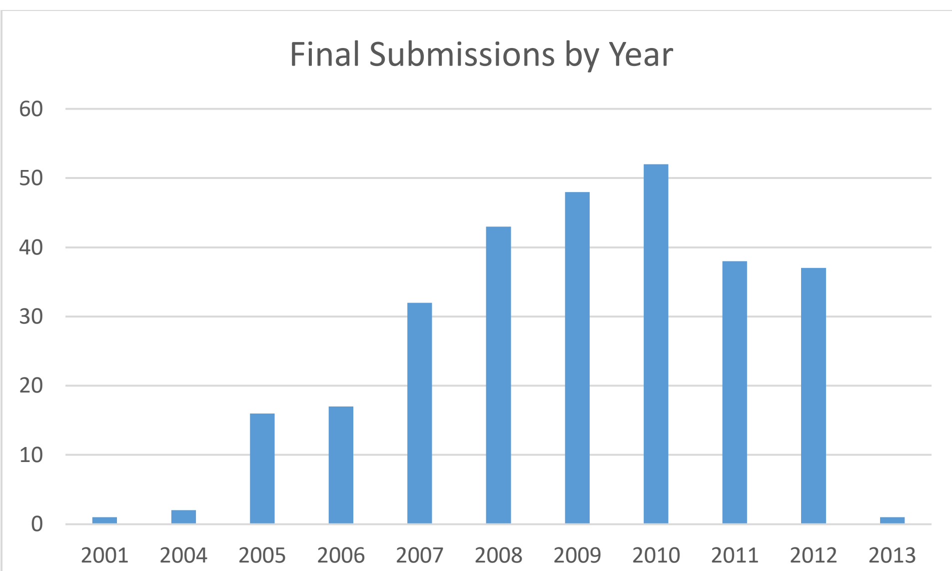


Figure 75: Data Age

Data age is important to consider when utilizing the resulting analysis The majority of the data used to generate this report was collected between 2004 and 2012. The relevance of historical data depends on how well the past represents future performance. In a DoD weapons systems environment, where the laws of physics govern many aspects of the software (e.g. avionics), historical data can remain relevant for quite a long time. On the other hand, AIS can be greatly influenced by COTS and the external environment (e.g., operating systems, cybersecurity, etc.), so the relevance of historical data needs to be balanced with how well the current environment resembles the historical software development environment.

Data Sharing

We have been granted permission to share all the data and source documents with the DoD cost community. Currently we use the AMRDEC SAFE Web Application (https://safe.amrdec.army.mil/safe/) to transfer these files securely. For information on obtaining the data and associated documentation, please contact the authors.