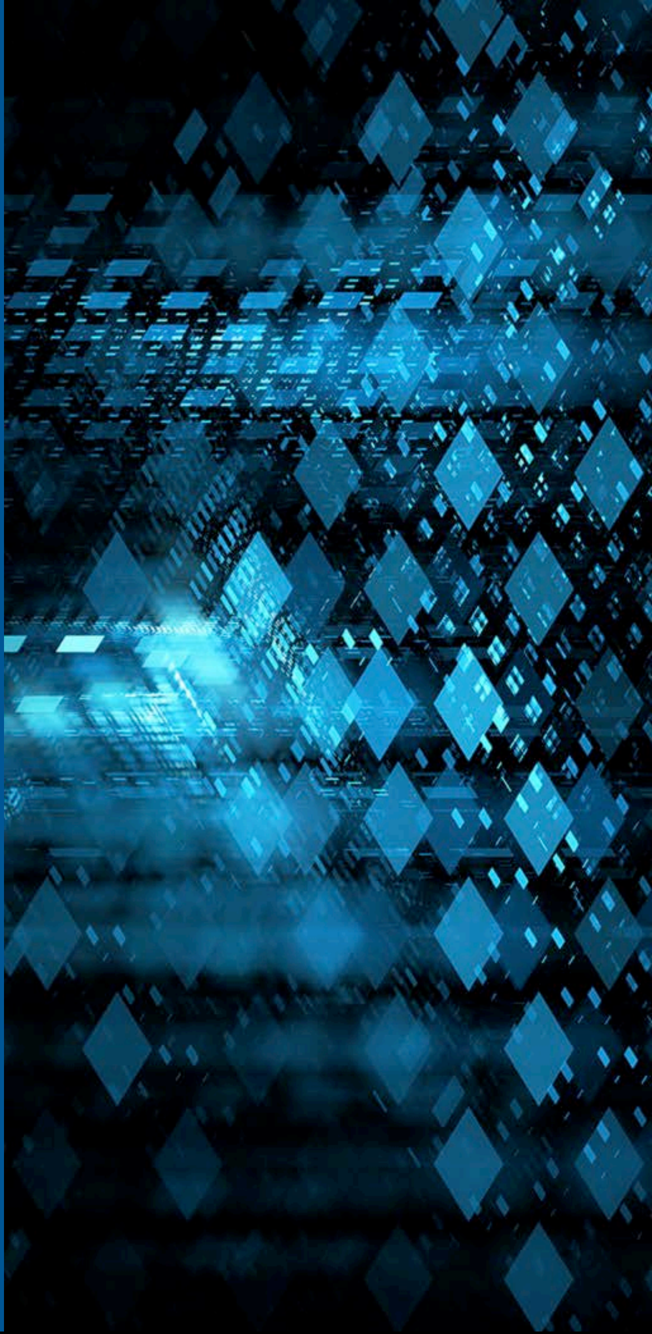
Exploring the DoD Software Factbook

Dr. Christopher Miler

Dr. Forrest Shull

Dr. David Zubrow

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

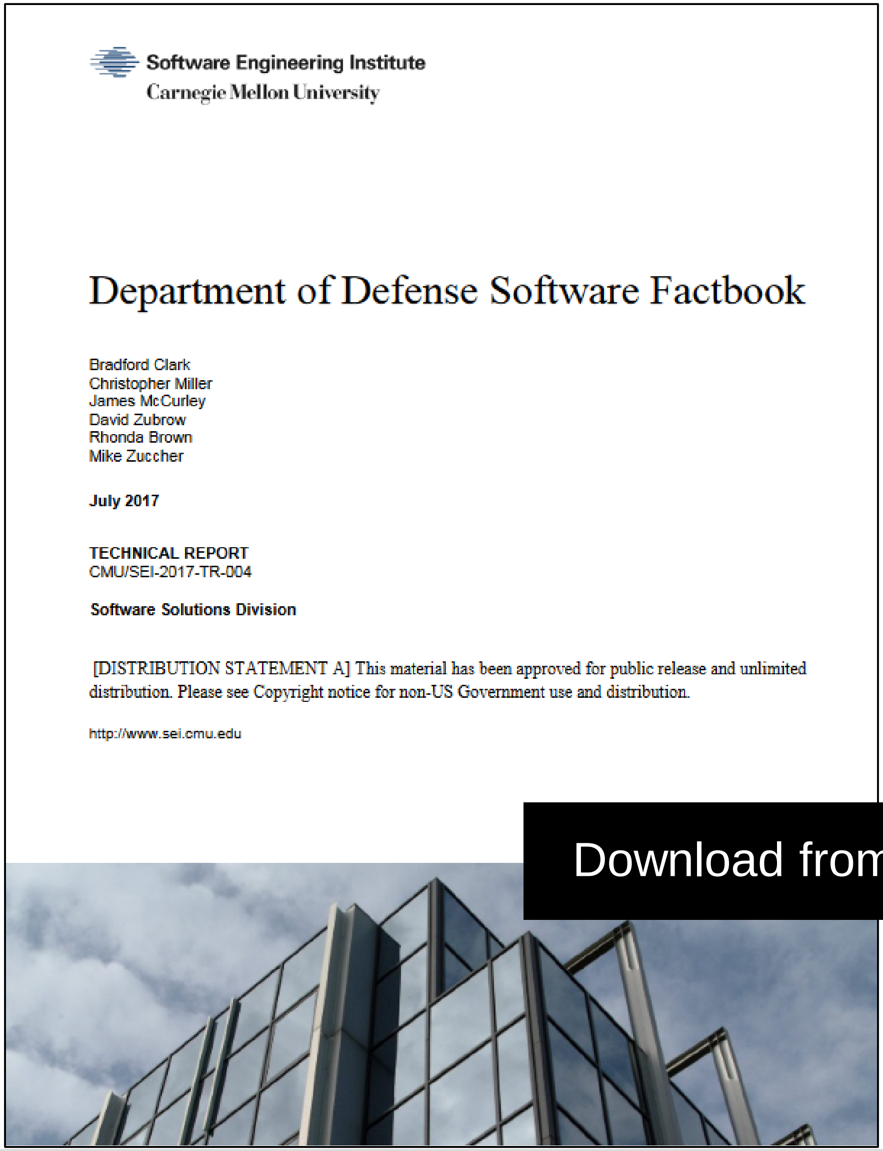


This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM18-0599

Understanding and Using DoD Data About Software



Software Engineering Institute
Carnegie Mellon University

Department of Defense Software Factbook

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

July 2017

TECHNICAL REPORT
CMU/SEI-2017-TR-004

Software Solutions Division

[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.

http://www.sei.cmu.edu

The Department of Defense (DoD) maintains an extensive collection of software engineering programmatic data.

The SEI analyzed this data, translated it into information that is frequently sought-after across the DoD, and published it in accessible form in a factbook.

Download from *http://resources.sei.cmu.edu/library*

DoD Factbook Contents

1 **How to Read this Document**

2 **Executive Summary**

3 **Introduction: DoD Software Projects 101 – Basic Facts**

- 3.1 Key Project Dimensions and Empirical Relationships
- 3.2 Functional Size (Requirements)
- 3.3 Product Size (ESLOC)
- 3.4 Effort
- 3.5 Duration
- 3.6 Team Size (People)
- 3.7 Productivity
- 3.8 Profile of a Typical Project

4 **Portfolio Performance**

- 4.1 Most and Least Expensive Software
- 4.2 Best-in-class/Worst-in-class

5 **Program Planning, Trade-offs and Risk**

- 5.1 Estimation Relationships
- 5.2 Effort-Schedule Trade off Analysis
- 5.3 Software Growth – Predicting Outcomes

6 **Conclusions and Next Steps**

Appendix A: Acronyms and Definitions

Appendix B: ESLOC

Appendix C: Super-domains

Appendix D: Operating Environments

Appendix E: Transforming Data

Appendix F: Predictive Models

Appendix H: Burden Labor Rate

Appendix J: Most-Least Expensive Software Analysis

Appendix K: Effort-Schedule Software Analysis

Appendix L: Best-in-class/Worst-in-class Software Analysis

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

Appendix N: Data Source Details

Download full report from *http://resources.sei.cmu.edu/library*

Distribution Statement A. This material has been approved for public release and unlimited distribution.

DoD Factbook 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 SEI analyzed and translated the data 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 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 questio**ns related to planning or replanning software projects.

Download full report from *http://resources.sei.cmu.edu/library*

Answering Common Questions

At the highest level, our analysis provides a general idea of how much a software ***project*** might cost and how long it might take.

- *How many requirements do DoD software projects usually have?*
- *How many lines of code do they contain?*
- *How many hours of work does it take to complete a software project?*
- *How long does a software project last?*
- *How many lines of code are produced per hour?*
- *How much do DoD software projects usually cost?*

About the Data

The data analyzed for the DoD Software Factbook comes from the Software Resources Data Report (SRDR).

The SRDR:

- is a contract data deliverable for formalized reporting of software metrics data
- is the primary source of data on software projects and their performance
- provides data at the project or subsystem level
- is used by all major contracts and subcontracts*
- records both estimates and actual results of new software or upgrades

*for contractors developing or producing software in 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

To be useful in our analysis, data had to include information about size (functional and product), effort, and schedule.

The data set we used for this analysis included “final report” data from 287 projects.

We used 181 pairs of initial and final cases for analysis of the estimated versus actual performance of projects.

Basic Benchmarks You Can Use

	Small projects	*Typical projects*	*Large projects*
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 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** million	**$3.3** million	**$8** million

*Based on an $82.24 hourly rate
Small projects are those at the 25th percentile and large projects are those at the 75th percentile.
The data set for this analysis used 287 projects from DoD SRDRs submitted by contractors for MDAP and MAIS projects.

Understanding the 3 Super Domains

Beyond basic benchmarks, findings in the Factbook are also presented by system type or ***super domain***.

Real-time (RT)

The most complex type of software, taking the most time and effort for a given system size due to the lower language levels, high level of abstraction, and increased complexity.

Examples

- Sensor control and signal processing
- Vehicle control
- Vehicle payload
- Real-time embedded

Engineering (ENG)

A software type of medium complexity that is generally executed on commercial off-the-shelf (COTS) software applications.

Examples

- Mission processing
- Executive, automation, and process control
- Scientific systems
- Telecommunications

Automated Information System (AIS)

This software automates information processing and allows a designated authority to exercise control over the accomplishment of the mission.

Examples

- Intelligence and information systems
- Software services
- Software applications

What We Learned: Some Key Results

- Software growth can be predicted from initial estimates.
- Real-time software is the most expensive software to develop, followed by engineering and automated information system software.
- Best-in-class software projects show significant gains in efficiency, speed, and cost reduction.

What We Learned, by Super Domain

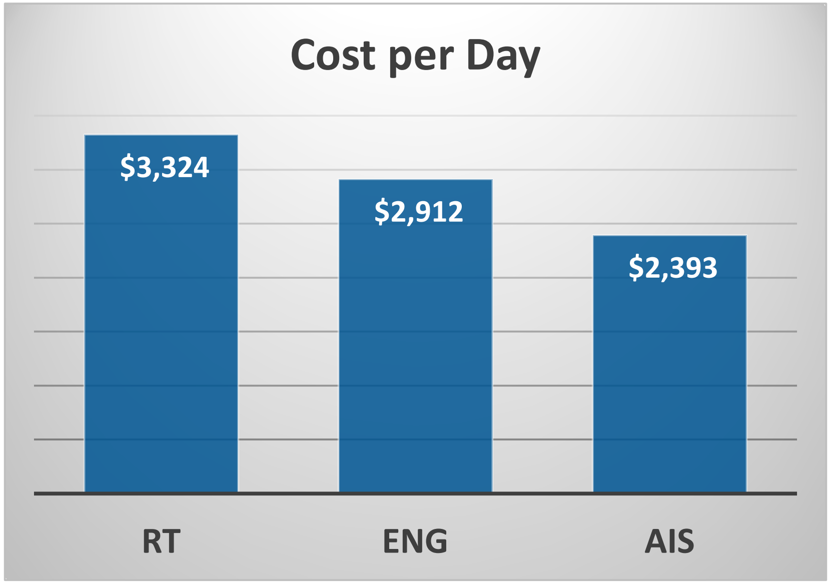
Software growth can be predicted from initial estimates.

- Initial estimates enable statistically strong predictions of the realized software requirements, size, effort, and schedule reported upon final delivery.
- 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).

What We Learned, by Super Domain

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

- Analysis revealed that real-time software costs 14% more to develop than engineering software, and 39% more than automated information system software.
- The typical cost per day for an typical-size project is $3,324 for real-time, $2,912 for engineering, and $2,393 for automated information systems.

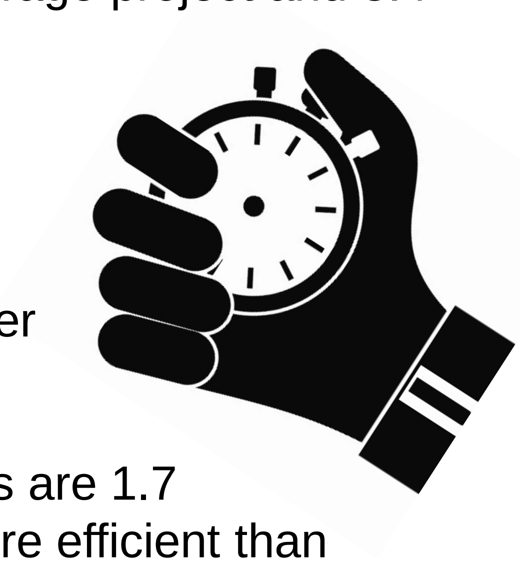


What We Learned, by Super Domain

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

- 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 **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.



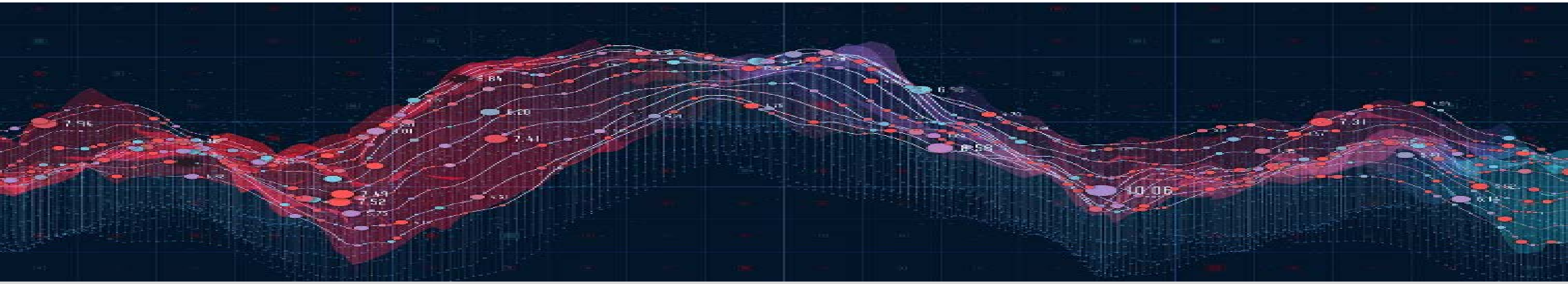
- The best-in-class **automated information system** projects are 1.7 times more efficient than average projects and 3 times more efficient than worst-in-class projects. Best-in-class projects are 2 times faster than average projects and 4 times faster than worst-in-class projects.

Project Planning, Trade-offs, and Risk

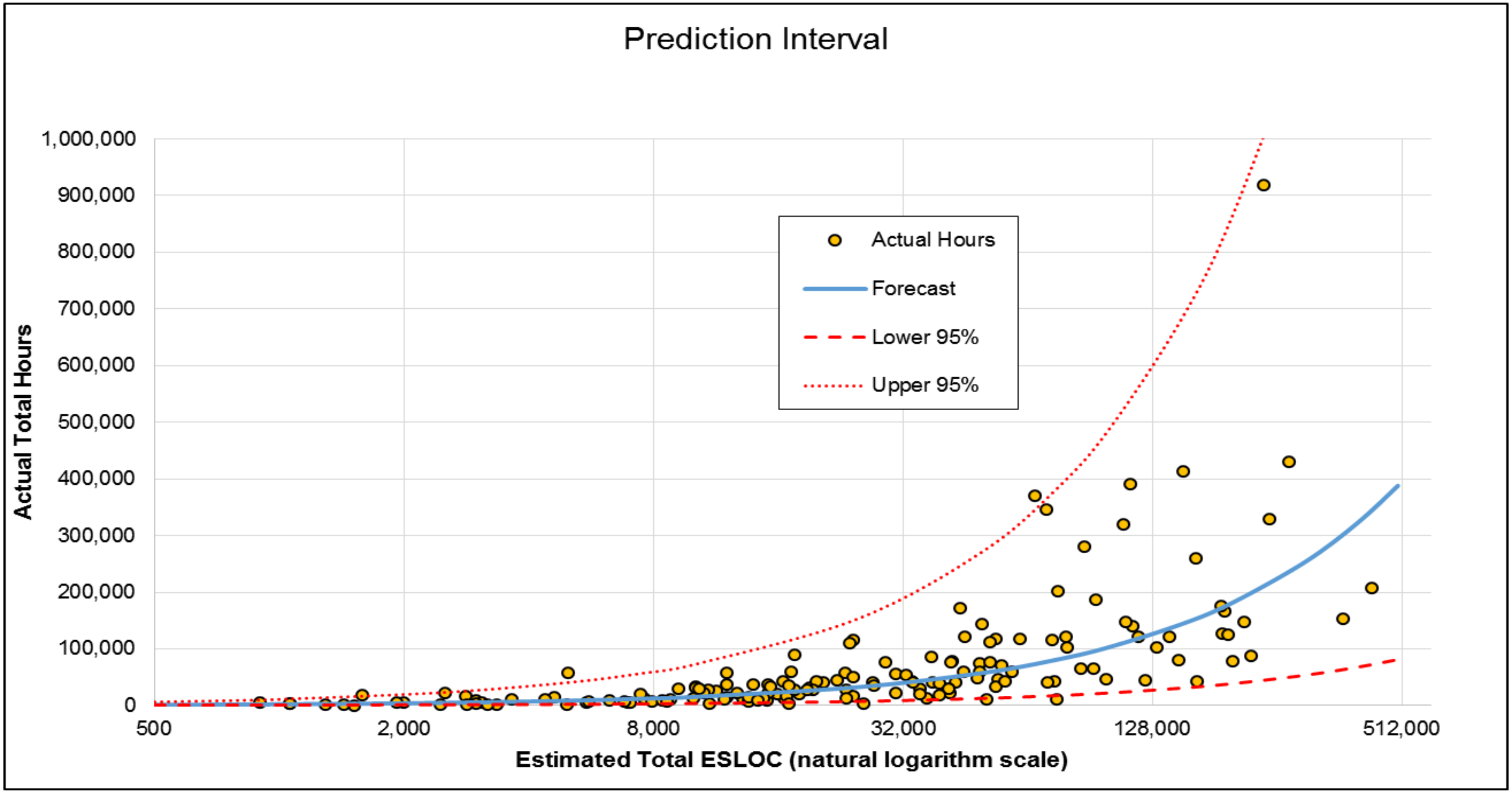
We conducted a more extensive analysis of the data, where we explored correlations among requirements, size, duration, and effort.

Here are the strongest models to emerge from this analysis.

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



Predicting Actual Total Effort by Estimated ESLOC (1)



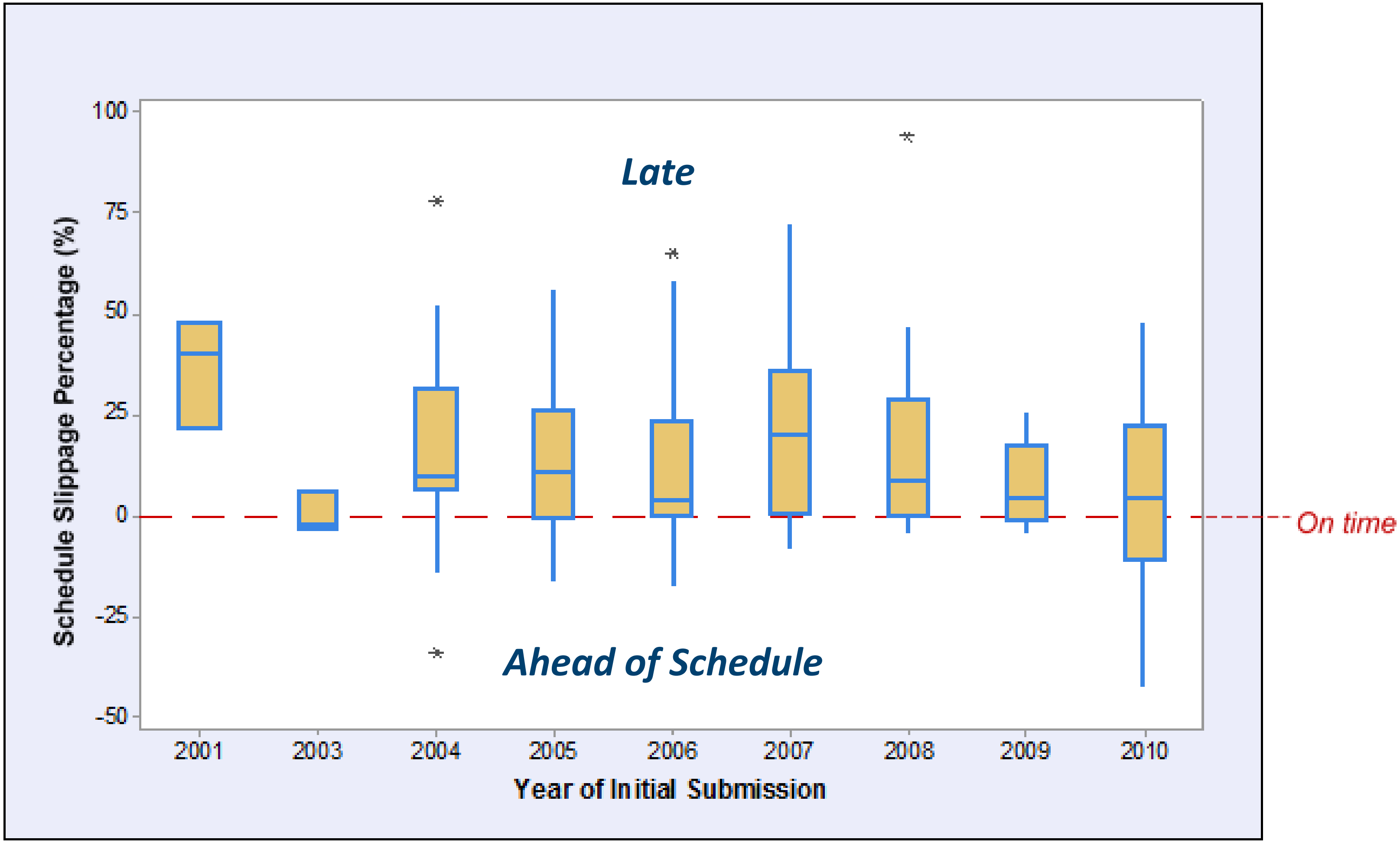
Predicting Actual Total Effort by Estimated ESLOC (2)

	Initial ESLOC Estimate	Forecast Total Hours	Percent difference (Actual & Estimate)	Prediction Interval – Total Hours	
				Lower 95%	Upper 95%
Under Estimated	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
+/- 10%	**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
Over Est.	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

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).

Software Growth – Predicting Outcomes

This represents the change in schedule, showing the difference between the estimated end dates from the initial submissions to the actual end dates reported in the final submissions.



Distribution Statement A. This material has been approved for public release and unlimited distribution.

Conclusions

The cost of software development varies depending on several factors.

- Different super domains have different levels of difficulty that cause more effort to be expended on more difficult software.
- 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.)

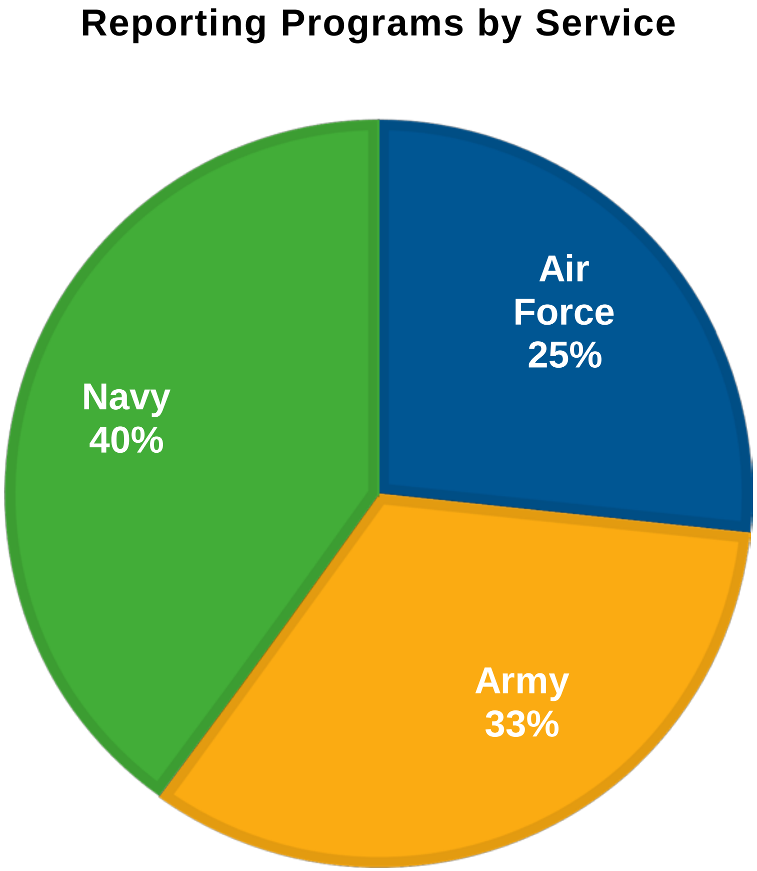
Distribution Statement A. This material has been approved for public release and unlimited distribution.

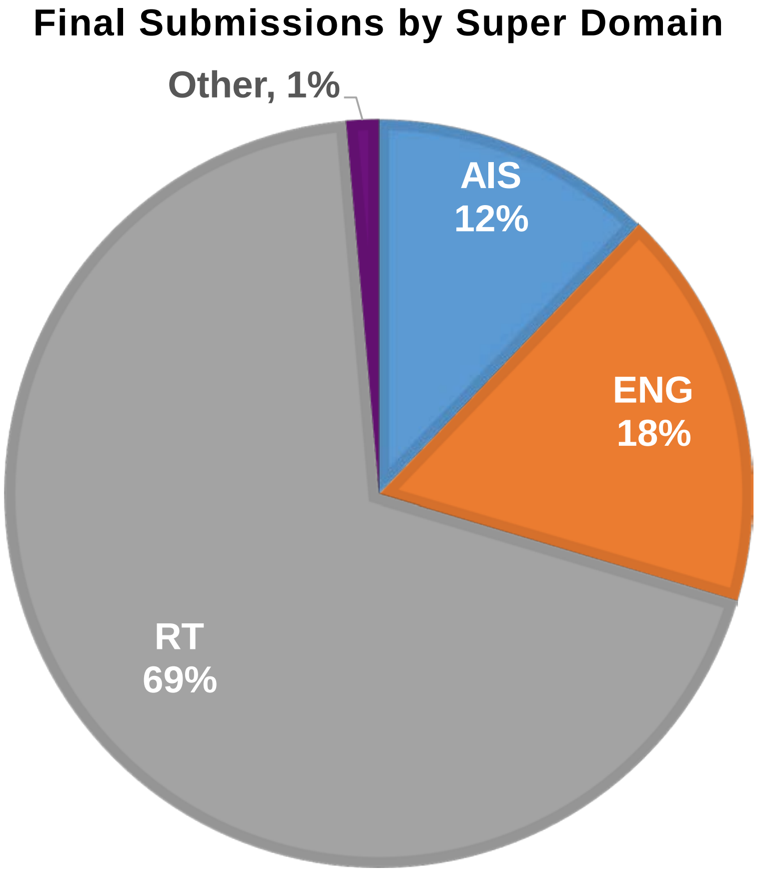
Future Plans

- Link the project data back to source documents and other data to investigate the data more fully
- Further investigate data to find out why best and worst projects perform differently
- Include additional SRDR data in the analysis to increase the fidelity of the super-domains groupings and provide a more robust analysis

For comments and suggestions, please contact:
fact-book@sei.cmu.edu

Data Details

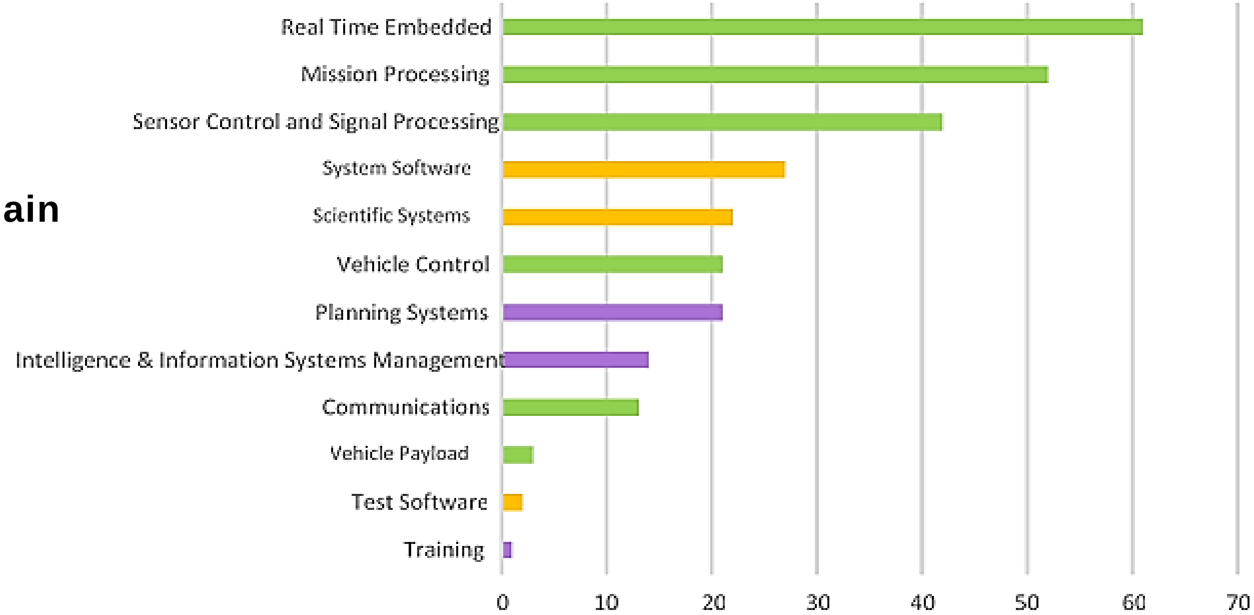




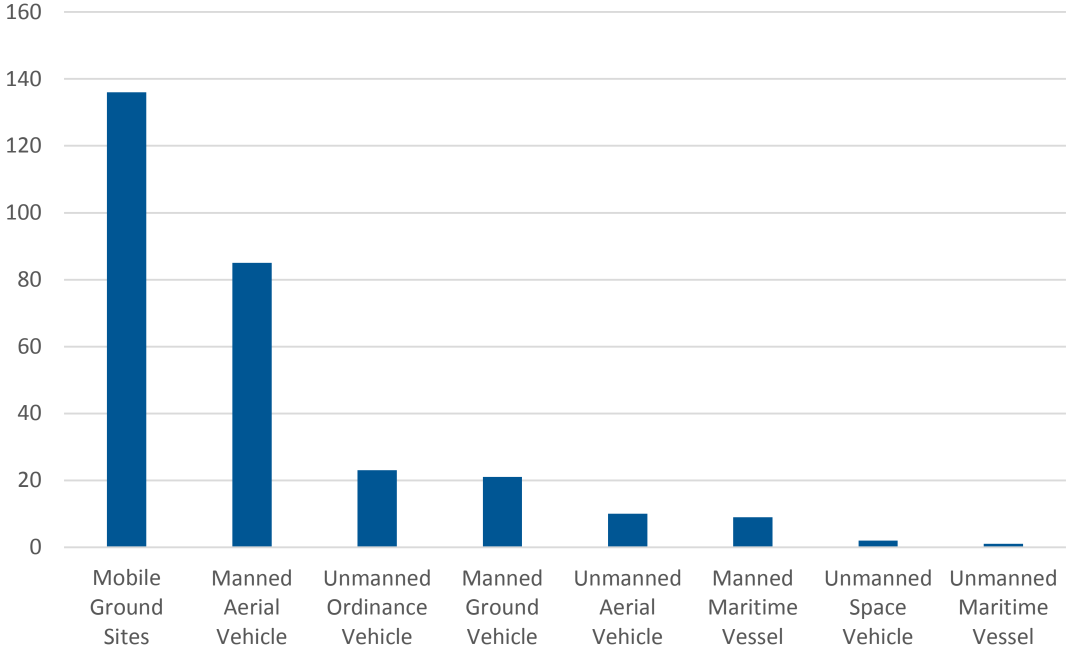
RT - Real-time
ENG - Engineering
AIS - Automated Information Systems

Data Details

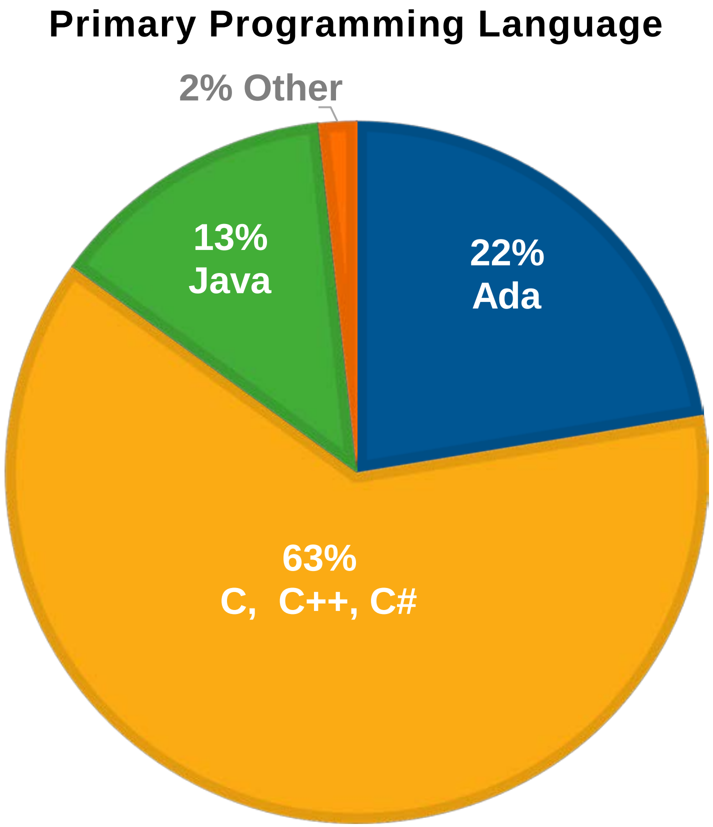
Program Distribution by Application and Super Domain

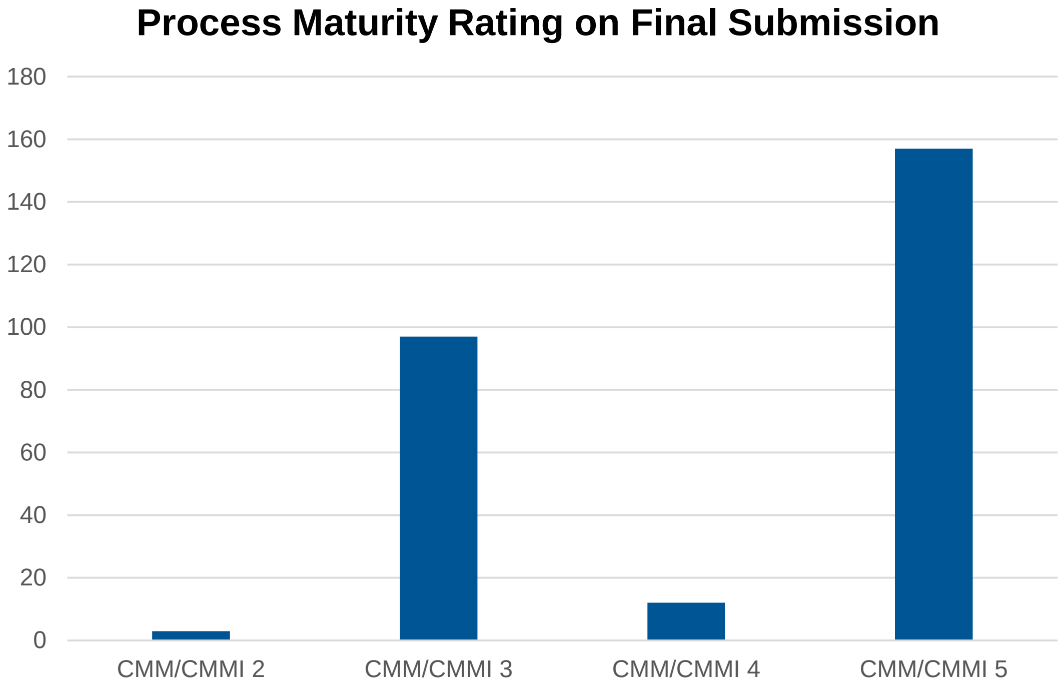


Final Data Submissions by Operating Environment



Data Details


Primary Programming Language
2% Other
Java
Ada
C, C++, C#


Process Maturity Rating on Final Submission
CMM/CMMI 2
CMM/CMMI 3
CMM/CMMI 4
CMM/CMMI 5