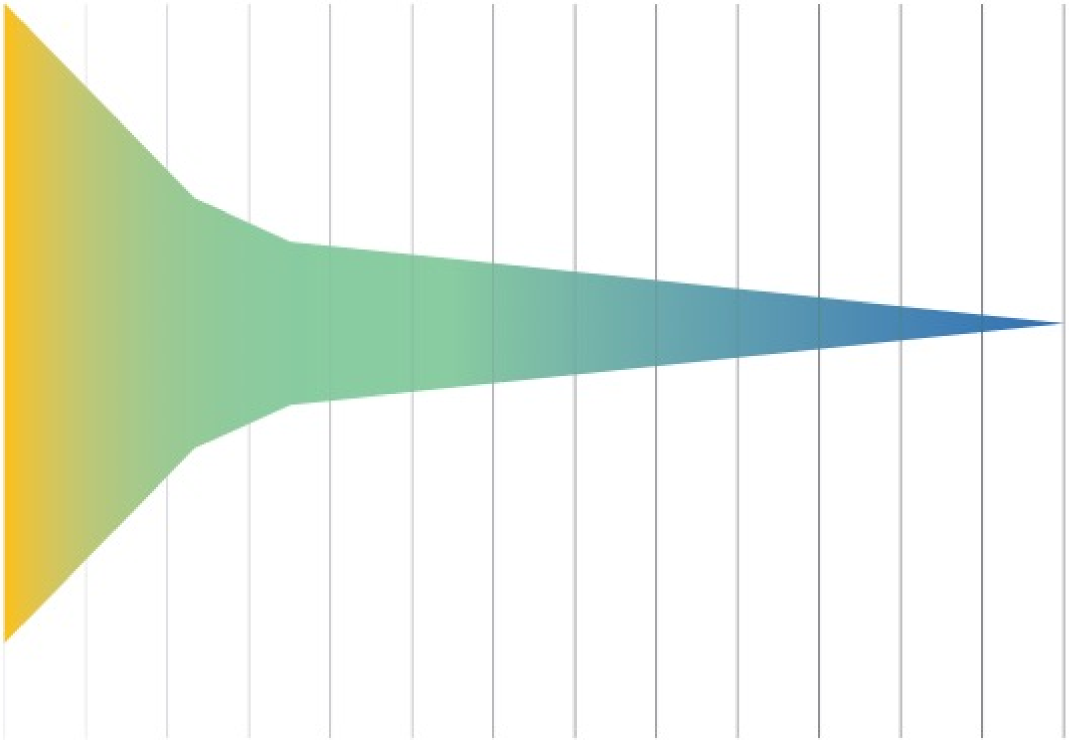


How to Produce Better Cost Estimates for Medium- to Large-Scale Software Development Projects

December 3, 2021

U.S. Department of Transportation
Volpe Center

> “We’ve run out of funding and the project is behind schedule…”

Introduction

Key takeaways from a summary of large-scale commercial and government IT projects produced by the Standish Group [1] is striking in that so few projects in the 10 years studied were completed successfully:

- Of 3,555 software projects that had labor costs of at least $10 million, only 6.4 percent were successful.
- 52 percent of the large projects were "challenged," meaning they were over budget, behind schedule, or did not meet user expectations.
- The remaining 41.4 percent were failures—they were either abandoned or started anew from scratch. [1]

The high-profile release of the HealthCare.gov website in October 2013 encountered immediate difficulties—including high website demand that caused the system to crash shortly after it launched and cost overruns. In a paper examining the challenges experienced with the website, it was noted that planned costs escalated from $93.7M to a final cost of $1.7B, and that the key causes of the cost increase were a lack of relevant software estimation experience, lack of technical leadership, and schedule pressure [2]. In a New York Times op-ed published after the HealthCare.gov website encountered challenges, technologists Clay Johnson and Harper Reed sought to explain why government almost never gets software tech right. [3] “Large federal information technology purchases have to end,” they stated while also noting that projects have a ”94 percent chance of failure or delay.” More recently, many states developed coronavirus vaccine registration websites that experienced performance issues and were later found to be poorly designed. [4]

This paper examines how best practices in software estimation can address some of these issues by ensuring that estimates for software development fully cover the scope of work and that initial schedules are realistic. Confidence levels associated with producing estimates for software development will be examined along with the importance of having a risk reserve. Other areas such as software development methods and how they may impact estimates will be examined, along with methods for producing software estimates. A checklist of important items is included to assist agencies, developers, and project managers achieve accurate software cost estimation. Additionally, this paper will identify how successful organizations manage to deliver transportation software-related projects on time, and within cost estimates.

Key Factors Associated with Software Estimation

For project managers, the process of estimating the cost to develop, test, and implement a large-scale software project can be daunting. However, if software estimators can break the problem down into smaller pieces and follow a number of steps in the process, the task can be considerably easier than imagined. Figure 1 illustrates the key factors that should be followed during the software estimation process.

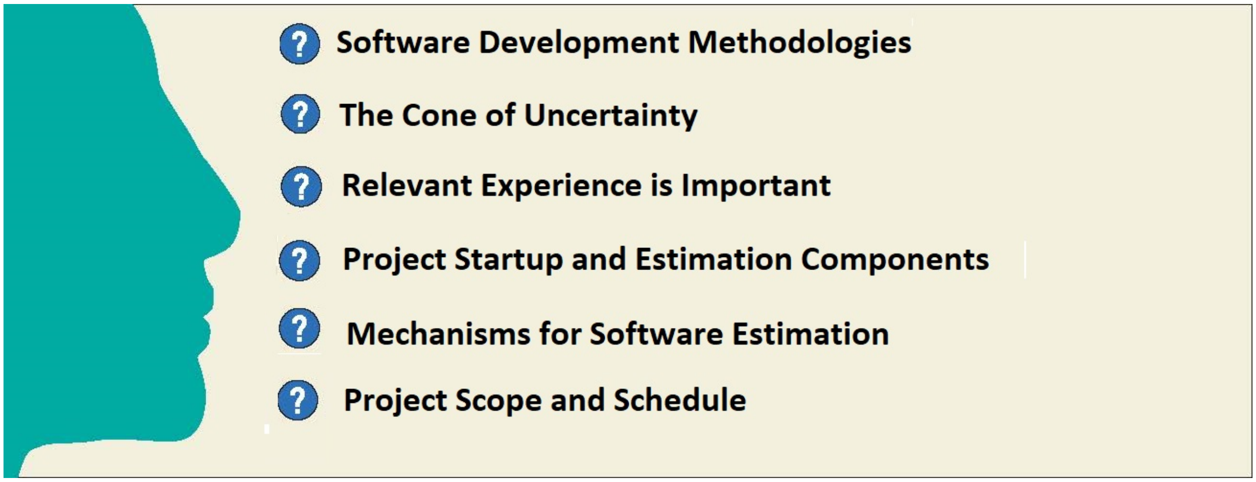

Software Development Methodologies
The Cone of Uncertainty
Relevant Experience is Important
Project Startup and Estimation Components
Mechanisms for Software Estimation
Project Scope and Schedule

Figure 1. The software estimation process involves many key factors

Software Development Methodologies

Software development methodologies are numerous and vary greatly. Four of the most common methodologies are:

Agile: The Agile software development method produces the final product in iterations with an emphasis on competent people working collaboratively. Agile focuses on rapidly developing working software that involves customer collaboration and seeks to respond to customer changes without being closely tied to a rigid plan. [5]

DevOps: DevOps development attempts to combine software development practices with the deployment and operations of the software. One of the goals of DevOps is to increase the frequency of software releases and streamline the deployment process. [6]

Waterfall: The Waterfall software development method breaks down the development process into a set of hierarchal steps or phases that are interconnected with each other. For large-scale government projects, this often requires that software requirements and design are fully defined before development activities can begin. [7]

Rapid Application Development (RAD): This method of software development breaks the process into phases that are focused on requirements planning, user interface design, software construction, and cutover (implementation). This process has been improved in recent years by adding prototype development and a focus on quality and risk. [8][9]

Understanding the differences in the focus of each methodology enables an estimator to think about the required activities and tasks, and related impact costs. For example, Agile development might incur travel-related costs for the customer to collaborate with on-site development teams. RAD development might require additional activities and costs to produce prototype software.

The Cone of Uncertainty

Perhaps the most important concept for developers and project managers related to software estimation is the level of confidence associated with an estimate. Figure 2 shows how software estimates can vary greatly based on the project phase with which they are associated. This diagram is commonly known as the Cone of Uncertainty [10], and is related to the conceptual phase or beginning of a project. The Cone of Uncertainty demonstrates that project cost estimates prepared in the early phase of a project may be incorrect by as much as a factor of four.

For agile projects, the unknown work represented in the Cone of Uncertainty shows up in the product backlog, as iterations proceed uncertainty is reduced. As we move through product definition and into the Requirements Complete stage, the ability to produce accurate cost estimates improves and the possible variance falls to 150 percent. It is for this reason many estimates are typically produced during the Requirements Complete stage. If we can move toward the Design Complete stage, the variance drops considerably to approximately 20 percent. Therefore, it is critical the estimator understands and communicates to the customer possible variances that can occur when producing estimates.

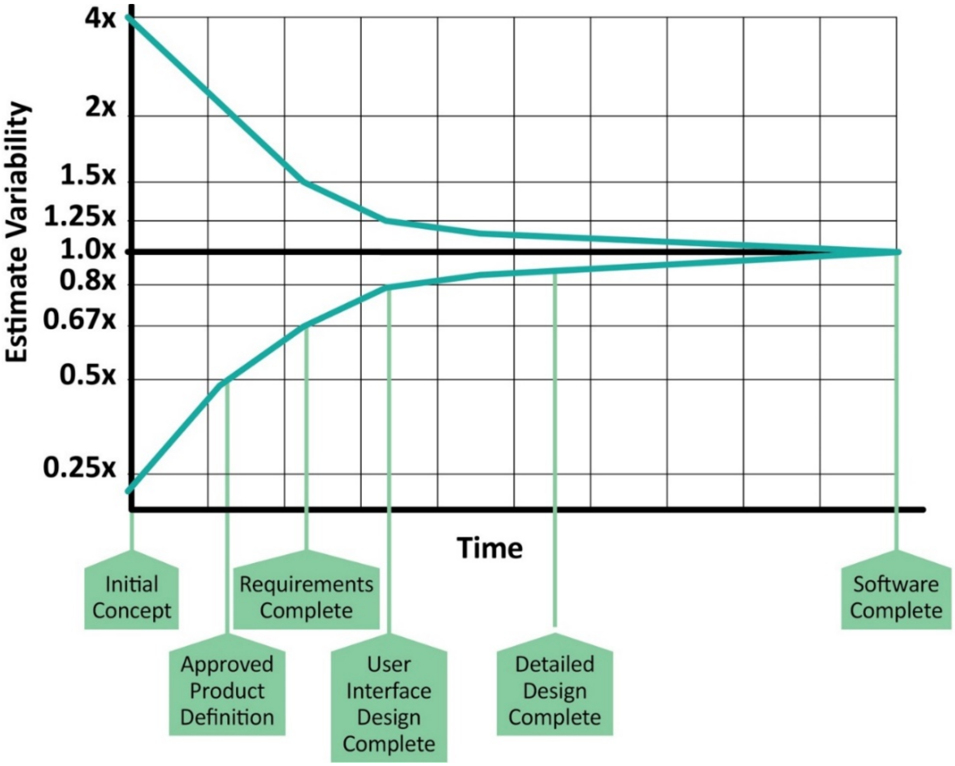


Figure 2. Cone of Uncertainty

One method that software estimators use to improve software estimation confidence levels is to develop a risk reserve. [11] This best practice sets aside funds in a separate account at the start of a project to address project risks. Risk reserves can be estimated by using Crystal Ball software that uses Monte Carlo simulation techniques to simulate the effect of identified project risks. [12] The risk reserve is typically managed by a Risk Management Committee in conjunction with a Configuration Control Board (CCB). Project risks are tracked by CCB members and the CCB allocates funds from the risk reserve to mitigate emergent issues. This requires that planned and actual project costs are tracked and that appropriate actions are taken before significant variances occur. In general, best practice tells us that projects should not be undertaken without a risk reserve.

Relevant Experience Is Important

Software project estimators should grow their skills by estimating small components of larger projects. As their estimating skills develop and improve over time, estimators can increase their skill set by estimating larger projects. Having a feedback loop involving estimators and stakeholders that compares actual project costs to the initial estimates is necessary for estimators to gain experience. Software project estimators should have subject matter expertise in the business domain and knowledge of the technical area for the project they are estimating.

Project Start-Up and Estimation Components

In the federal government, a project sponsor will approach a development organization with a vision of a system that will help solve a business or mission-oriented problem. This vision is often high level and captured briefly in a few PowerPoint slides or a white paper. The sponsor often has modest start-up funding available and may already have a potential cost in mind. At this point, it is important for the sponsoring and performing organizations to develop a clear and common understanding of available budget and project estimates. Additionally, since the process of planning and budgeting for projects in government can take many years, the task of matching available budgets to the estimate can be very challenging [13], which further underscores the need for good project cost estimates.

Before addressing methods to produce a good estimate, it is critical to consider the entire scope of the system that will be estimated. It is important to break down the system into various components or software modules. It is also important to document and estimate costs for interfacing with other systems. System interface costs can be significant as these may involve other development groups and require significant coordination. Areas that are typically overlooked include reporting, algorithms, implementation of user roles, online help, and documentation. Estimation of security costs should not be overlooked as these can be considerable. The National Institute of Standards and Technology (NIST) Publication 800-53 [14] outlines over 1,000 security controls that should be considered when designing and implementing information systems. Hence, it is crucial to include the costs to build required security controls into the system and also for conducting security testing.

Other non-functional requirements such as performance, maintainability, reliability, scalability, and usability also need to be estimated. Estimators also need to think about costs for ongoing maintenance, deployment, user training, and education.

Not only do we have to break down the product into its subcomponents, but we also need to think about the required effort during the life cycle of the development process as we move from needs analysis to design, development, testing, and deployment. The project estimator should consider the following important questions:

- How many testing events will be required?
- Will developers conduct remote or in-person testing?
- Will travel to and from sites be required when the system is deployed?

Mechanisms for Software Estimation

Software estimates can be produced in several ways. [15] Typically, these include analogous estimation, bottom-up, 3-Point, Constructive Cost Model-based tools, and commercial software estimation products such as QSM SLIM-Estimate and SEER-SEM [16] (Software Evaluation and Estimation of Resources - Software Estimating Model).

Analogous estimation is best for smaller projects (< $700K) where a similar project was recently completed. The software estimator would start with the actual costs of a similar software project, and then examine the differences between the two projects to produce an estimate. Estimates produced in this manner can often be quite accurate, as long as the differences are small, and all the factors are considered. In such cases, the Cone of Uncertainty is much smaller as most of the functionality has already been implemented in a similar project.

Another popular approach for software cost estimation is the bottoms-up method. In this method, software developers create detailed estimates for each sub-component of the system. Since software developers are detailed-oriented, this method can be very time consuming and requires the design to be well elaborated to at least the screen design level. A challenge here can be that detail-oriented developers struggle with the ambiguity of developing estimates, so this task may be better performed by technical team leads.

3-Point estimation techniques calculate optimistic, pessimistic, and most likely estimates for a project. These values are calculated for each task and then combined to produce an overall expected (most likely) estimate along with best-case and worst-case scenarios. Estimators can use this technique in many ways, but it really helps them understand the possible outcomes that could occur as the project moves ahead.

For large-scale projects (multiyear, multimillion dollar), estimators can use commercially available tools such as SEER-SEM to produce more accurate estimates. SEER-SEM primarily uses ***functional size*** [17] as a mechanism to estimate level of effort. Functional size is a unit of measure that represents the effort needed to produce an individual functional user requirement for a system. SEER-SEM and other commercial products generally provide many parameters the estimator can use to indicate project stage, knowledge of developers, level of security, etc. These parameters along with an extensive built-in project knowledge base are used in conjunction with the function points to calculate effort in terms of hours and cost. One of the shortfalls of SEER-SEM is that the estimates can be overly sensitive to the parameters. For example, setting the security requirements parameter to a very high level can increase the estimate by as much as 200 percent.

Project Scope, Schedule, and Other Considerations

Other factors that affect the final project cost are the mechanisms put in place to manage project scope. Producing a detailed cost estimate helps project managers control project scope as all of the work has been estimated to a sufficient level of detail. If project managers or developers suggest changes, they can be evaluated to determine whether they are within the project scope. If the changes are beyond the project scope, additional funds will need to be added or un-coded items removed from the project. For example, many state vaccination websites were overwhelmed with traffic, so some agencies added a virtual waiting room where users could wait before going to the main website to register. Adding a waiting room would likely be outside of the original scope and cost estimate, and hence considered a change.

Estimation by itself is insufficient without a sound project schedule. It is not enough to simply divide the amount of work by the available resources to calculate a plausible project duration. Dependencies exist between tasks that can cause the overall duration to lengthen, which in turn increases the cost of the project. The U.S. DOT Volpe Center has found that for multimillion-dollar development projects, a duration based solely on effort and resource availability must be adjusted by a factor of 1.3 to produce a realistic schedule. Moreover, the mere act of thinking about tasks almost always turns up more project work that must be estimated. This type of orthogonal thinking is key to developing better estimates, since looking at the same problem from multiple perspectives will produce new insights. These insights can help improve the estimate.

For multimillion dollar software projects, Volpe experts recommend a duration based solely on effort and resource availability should be adjusted by **a factor of 1.3** to produce a realistic schedule.

Another best practice associated with producing project schedules is to estimate the cost to complete the project at a given point in the software development cycle (e.g., when the schedule says that development is 50 percent complete). An estimator would re-estimate the cost of the yet unfinished work. Findings can be used to help determine if the project will meet its budget objectives or not, and to understand areas where the original estimate was lacking. As part of this re-estimation process, team members and software estimators should brainstorm to ensure that all the remaining tasks are included.

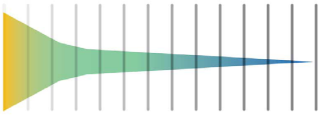
Keeping a database of project estimates and actual costs is another useful method that estimators can use to learn from mistakes made with previous estimates. Additionally, this allows “Corporate Knowledge” associated with developing estimates to be retained.

Estimating transportation-related software projects in the government space poses additional challenges, as software tends to be more complex due to advanced logic that is needed to implement business rules related to government regulation. Data quality is another important part of software projects, and federal technology experts should consider the need to standardize data when estimating large-scale software development projects. [18] Addressing data quality can be costly, [19] as many federal systems in the past did not validate data entered against standard data values (e.g., U.S. state abbreviations). When these are modified or merged with new system data values, significant costs can be encountered.

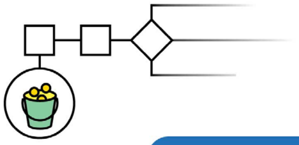
Finally, estimation must also consider the cost escalation that occurs over time as the price of resources can increase every year due to inflation. For multiyear projects, these cost escalations can be significant, for example a 3 percent increase in costs each year over a 5-year period will result in a 12.6 percent increase in costs in year 5.

CHECKLIST FOR ESTIMATING SOFTWARE DEVELOPMENT PROJECTS

□ **1.** Elaborate designs as far down **the Cone of Uncertainty** as possible, even if this means making a few assumptions.

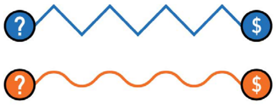


□ **2.** Before initiating a project, have a **risk reserve** and a method in place for managing changes in project scope.

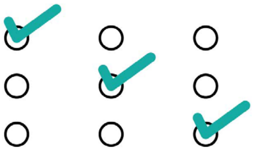


□ **3.** Ensure **sponsors fully understand** the project's scope and the methods that were used to produce the software estimates.

□ **4.** Always use **at least two** methods to estimate project cost and remember to compare the estimates generated by each method.

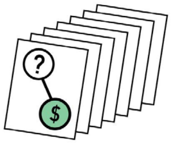


□ **5.** When using SEER-SEM or similar products, it is especially important to calibrate SEER model parameters by comparing SEER output with estimates produced using other methods to **ensure that SEER model parameters are set correctly.**

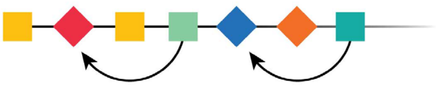


□ **6.** Develop a realistic **project schedule** in conjunction with the cost estimate.

□ **7.** Keep a **database of actual costs and estimates** from previously estimated projects.



□ **8.** If possible, **re-estimate project costs** during the software development life cycle.



U.S. Department of Transportation
Volpe Center

Conclusion

In summary, what can government agencies do to improve their cost estimates for large-scale projects?

1. Elaborate designs as far down the Cone of Uncertainty as possible, even if this means making a few assumptions.
2. Ensure that a risk reserve is included in project estimates.
3. Ensure sponsors fully understand the project’s scope and the methods used to produce the software estimates.
4. Always use at least two methods to estimate project costs and remember to compare the estimates generated by each method.
5. When using SEER-SEM or similar products, it is especially important to calibrate SEER model parameters by comparing SEER output with estimates produced using other methods to ensure SEER model parameters are set correctly.
6. Develop a realistic project schedule in conjunction with the cost estimate.
7. Keep a database of actual costs and estimates from previously estimated projects.
8. If possible, re-estimate project costs during the software development life cycle

Looking forward, software estimation techniques must evolve as automation becomes increasingly sophisticated. In the not-too-distant future, the transportation domain will see machine learning and other artificial intelligence (AI) approaches integrated into vehicles, risk models, data analysis, and safety management systems. Software developers will need to estimate automation projects supporting advances in autonomous vehicles, vehicle-to-vehicle communications, transportation equity models, climate change simulations, and other transportation-related innovations. It goes without saying that AI and Machine Learning will play a large role in how we estimate software costs in the future.

References

[1] Thibodeau, P. Healthcare.gov website ‘didn’t have a chance in hell.’ *Computer World*. October 21, 2013. https://www.computerworld.com/article/2486426/healthcare-gov-website--didn-t-have-a-chance-in-hell-.html.

[2] ABC123 Alumni. The failed launch of www.HealthCare.gov. HBS Digital Archive. October 18, 2016. https://digital.hbs.edu/platform-rctom/submission/the-failed-launch-of-www-healthcare-gov/

[3] Johnson C., Reed H. Why the government never gets tech right. *The New York Times*. October 24, 2013. https://www.nytimes.com/2013/10/25/opinion/getting-to-the-bottom-of-healthcaregovs-flop.html

[4] Keegan, J., Lacher, C. We ran tests on Every States COVID-19 Vaccine Website. The MarkUp. March 24, 2021. https://themarkup.org/blacklight/2021/03/24/we-ran-tests-on-every-states-covid-19-vaccine-website

[5] Beck, K., et al. (2001). The agile manifesto. Agile Alliance. http://agilemanifesto.org/

[6] Kim, G. (2018). The Phoenix Project (a novel about IT, DevOps, and helping your business win). IT Revolution Press.

[7] Royce, W. (1970). Managing the development of large software systems. Proceedings of IEEE WESCON.

[8] Iivari, J. Hierarchical spiral model for information system and software development. Part 1: Theoretical background. Information and Software Technology, 32(6), 1990. https://doi.org/10.1016/0950-5849(90)90125-B

[9] Kerr, J., Hunter, R. (1993). Inside RAD: how to build a fully functional system in 90 days or less. McGraw-Hill. ISBN 0-07-034223-7.

[10] Boehm, B. (1981). Software engineering economics. Prentice-Hall.

[11] Murray, J. Ramsaur, W. Project reserves: a key to managing cost risks. *Project Management Quarterly*, 14(3), 71–77, 1993.

[12] Clark, K. Basic cost Risk Analysis: Using Crystal Ball on government life cycle cost estimates. Proceedings of the 2007 Crystal Ball User Conference.

[13] Makai, M., Why Does Building Software for the Government Cost So Much? CodingAcrossAmerica, 2012-2017. https://www.codingacrossamerica.com/why-does-building-software-for-government-cost-so-much.html

[14] Security and Privacy Controls for Information Systems and Organizations, NIST Special Publication 800-53, Revision 5 https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/archive/2020-03-16

[15] McConnell, S. (2006). Software estimation: demystifying the black art (developer best practices). Microsoft Press.

[16] SEER for software user's guide. (2014). Galorath Incorporated.

[17] IFPUG counting practices manual, release 4.3. (2010). Princeton Junction, New Jersey: IFPUG.

[18] Richard Y. Wang & Diane M. Strong. (1996). Beyond accuracy: what data quality means to data consumers. Journal of Management Information Systems, 12:4, 5-33.

[19] Haug, A., Zachariassen, F., & Liempd, D. (2011). The costs of poor data quality. Journal of Industrial Engineering and Management, 4(2), 168-193. http://dx.doi.org/10.3926/jiem..v4n2.p168-193

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering, and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE

December 2021

3. REPORT TYPE AND DATES COVERED

Final Report

4. TITLE AND SUBTITLE

How to Produce Better Cost Estimates for Medium to Large-Scale Software Development Projects

5a. FUNDING NUMBERS

6. AUTHOR(S)

Edwin Tait, U.S. DOT Volpe Center, Aviation Safety Management Systems

5b. CONTRACT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Office of Strategic Initiatives for Research and Innovation
U.S. Department of Transportation
John A. Volpe National Transportation Systems Center
55 Broadway
Cambridge, MA 02142-1093

8. PERFORMING ORGANIZATION REPORT NUMBER

DOT-VNTSC-22-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Department of Transportation
John A. Volpe National Transportation Systems Center
55 Broadway
Cambridge, MA 02142-1093

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

Prepared under the direction of Regina Houston, Chief, Aviation Safety Management Systems and Ellen Bell, Director of Strategic Initiatives for Research and Innovation.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

This document is available to the public on the U.S. DOT Volpe Center website:
https://www.volpe.dot.gov.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Estimating transportation-related software projects in the government space presents specific challenges, as software tends to be more complex due to advanced logic that is needed to implement business rules related to government regulation. The U.S. DOT Volpe Center has found that for multimillion-dollar software development projects, a duration based solely on effort and resource availability must be adjusted by a factor of 1.3 to produce a realistic schedule. Moreover, the mere act of thinking about tasks almost always turns up more project work that must be estimated. This type of orthogonal thinking is key to developing better estimates, since looking at the same problem from multiple perspectives will produce new insights. These insights can help improve the overall development estimate and project scope and schedule.

14. SUBJECT TERMS

U.S. DOT, Volpe National Transportation Systems Center, IT projects, software development, medium-scale projects, large-scale projects, best practices, software estimation, software estimators, software development methodologies, cone of uncertainty, project scope, project schedule

15. NUMBER OF PAGES

12

16. PRICE CODE

NA

17. SECURITY CLASSIFICATION OF REPORT

None

18. SECURITY CLASSIFICATION OF THIS PAGE

None

19. SECURITY CLASSIFICATION OF ABSTRACT

None

20. LIMITATION OF ABSTRACT

Unlimited