



Sizing Software Modernization; A Real-World Example



Cost Estimating Team

- NAVAIR
 - Nicholas Taw
- NSI
 - Chad Lucas

Agenda

- Problem Statement
- About TMPC
- The Case for Modernization
- What was known
- The Estimation Plan
- Executing the Estimation Plan
- Estimate Results
- Program Benefits
- Path Forward
- Conclusions and Lessons Learned

Problem Statement

- Theatre Mission Planning Center (TMPC) is going through a modernization effort.
- A cost estimate was required to justify funding requests for the effort.
- Previous efforts at estimating the costs of modernization (using SLOC) resulted in irrational results.
- Previous estimates were not adjustable without input from the developers.
- Previous estimates were not useful as program management tools.
- A new approach to conducting the estimate was needed.

About TMPC

- Theatre Mission Planning Center (TMPC) is a software-intensive system of systems
- For the Tomahawk cruise missile and the Conventional Prompt Strike (CPS) weapon.
 - TMPC has been installed at over 180 sites globally.

About TMPC (cont)

- TMPC has three developers:
 - One for, Weaponing (designing and attack with weapons) and Navigation tools and services.
 - Another for the Tomahawk Mission Planning tools and services.
 - A third for the Task Management, Strike Planning & Strike Execution Software and other related tools and services.
- A fourth contractor acts as a prime integrator.

The Case for Modernization

- The current TMPC software architecture was designed in line with industry best practices of 25 to 40 years ago, before the existence of modern cybersecurity, software engineering techniques, and human systems integration (HSI) practices.
- Redesigning to modernize the architecture of TMPC implements DoD-mandated HSI and NSA-required cyber standards, streamlines “zero-trust” operations, and enables continuous development/delivery system upgrades.
- TMPC is moving from an incremental delivery approach to an Agile development methodology.

What was Known

- Redesign will result in coding new and existing capabilities in micro-services versus the current large component structure.
 - Micro-services is composed of small independent services that communicate across the Application Programming Interface (API), reducing code interdependencies.
- These micro-services would be “containerized” to allow for easier and faster updates in the future.
 - To minimize risk to existing operations and reduce the risk of failure the process of software strangler migration will be used.

What was Known

- Software Strangler Migration “is a software design and architectural pattern used in the context of legacy system modernization or application migration. The pattern’s name comes from the way certain plants, like strangler figs, grow around host trees, eventually replacing or “strangling” them entirely. Similarly, the Strangler Pattern aims to replace or evolve an existing system gradually, without causing any significant disruptions to the overall system functionality.” For more information on the Strangler Pattern, you can visit:

<https://www.techtarget.com/searchapparchitecture/tip/A-detailed-intro-to-the-strangler-pattern>

What was Known

- Many design decisions had not yet been made and full scope of what a containerized solution would look like had not been flushed out.
- The DevSecOps implementation plan is to have an iterative rollout, which the DevSecOps environment could change once the Continuous Authority to Operation (CATO) is developed, cloud infrastructure is developed, full Agile is embraced, and DoD Software Factories are established.
- Programming languages to be used had been previously identified.

The Estimation Plan

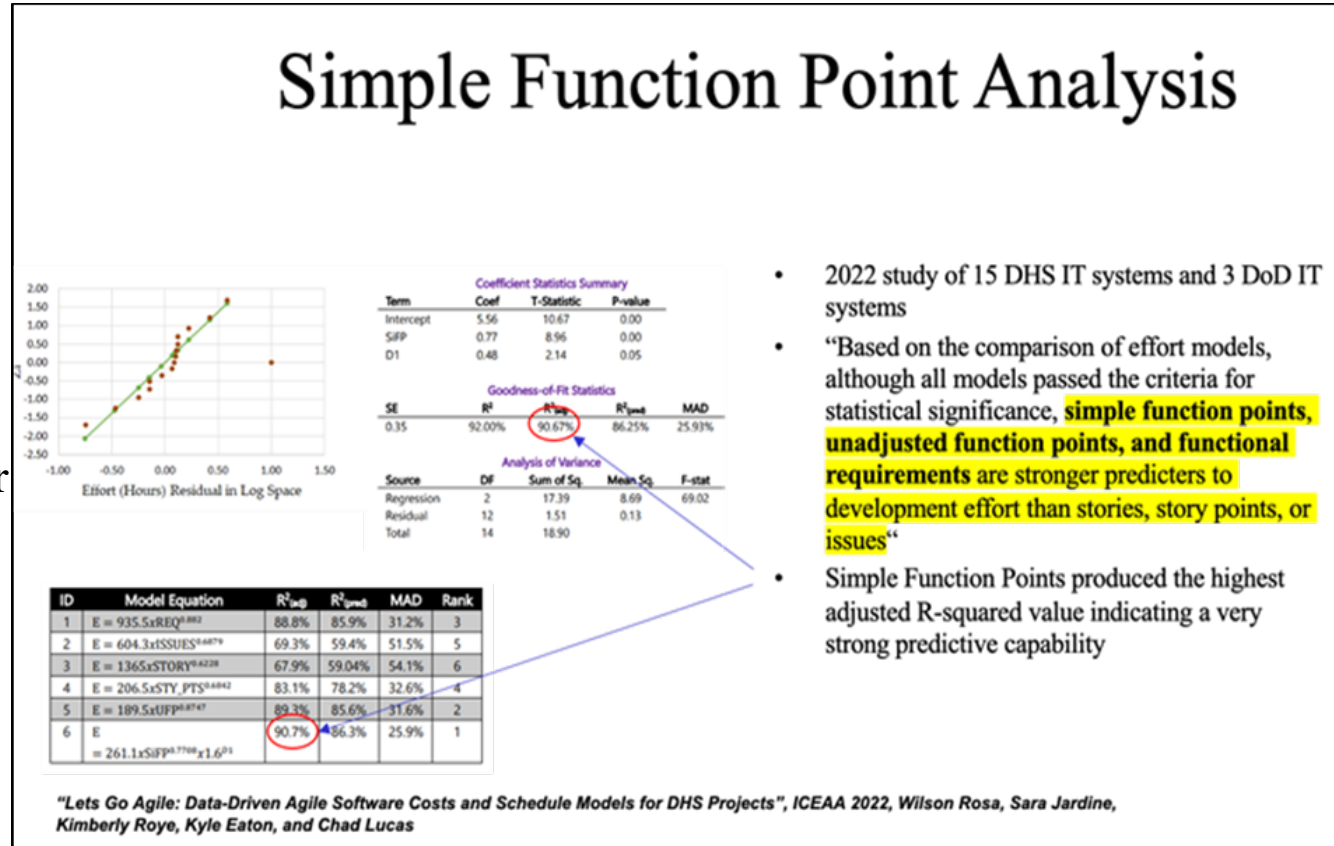
- Requirements for the system “as is” could be used as a baseline for sizing.
- Some level of reuse would have to be accounted for, but all requirements will have some effort required to complete the redesign.
- Simple Function Points (SFP) would provide the best path to an objective sizing measure and in a timely fashion.
- Assumptions would be informed by John’s Hopkins Applied Physics Lab (JHU APL) who, as the lead engineer, had been tasked with completing a comprehensive study.

The Estimation Plan (cont)

- Simple Function points:

- Are an object measure of size easily applied to programs in their early phases of development.
- The counting conventions around SFP make it easy for the counts to be validated.

- [Introducing Simple Function Points \(SFP\) - IFPUG - International Function Points Users Group](#)



- 2022 study of 15 DHS IT systems and 3 DoD IT systems
- “Based on the comparison of effort models, although all models passed the criteria for statistical significance, **simple function points, unadjusted function points, and functional requirements** are stronger predictors to development effort than stories, story points, or issues⁴⁴
- Simple Function Points produced the highest adjusted R-squared value indicating a very strong predictive capability

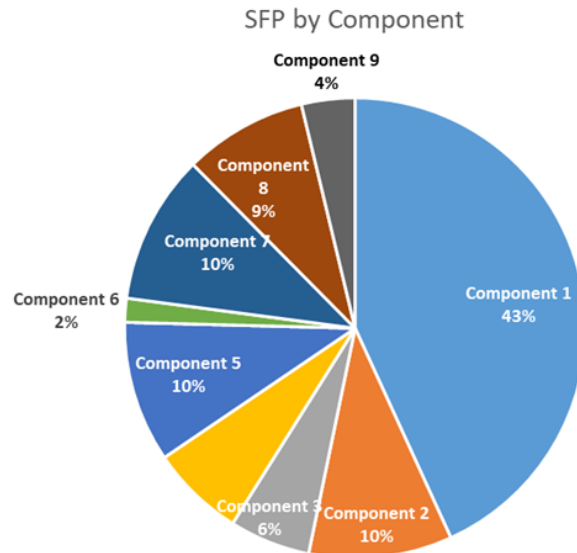
Executing the Estimation Plan

- A full listing of the System/Subsystem Specifications (SSS) by component was obtained from the prime integrator. SFP counts were based on these SSS statements.
- The cost team engaged the systems engineering team for help in validating the assumptions used and verifying whether duplicate requirements were in fact duplicates.
- The engineering team also spot-checked the actual counts to ensure the cost team had full understanding of the requirements as written.
- Result was a valid SFP count that could be used as basis for estimating the development effort.

Executing the Estimation Plan – Sizing the Effort

Simple Function Point Count Results

	Component 1	Component 2	Component 3	Component 4	Component 5	Component 6	Component 7	Component 8	Component 9	Total
Total Functional Requirements	1,637	197	154	244	289	56	340	262	93	3,272
<i>Non-functional listed as functional</i>	185	6	14	44	17	1	8	0	0	275
<i>Duplicate Requirements</i>	123	4	17	10	20	0	9	3	3	189
Actual Functional Requirements	1,329	187	123	190	252	55	323	259	90	2,808
Total EPs	1,455	336	179	230	326	60	339	294	118	3,337
Total LFs	236	57	39	31	59	7	67	46	26	568
Total SFP	8,345.0	1,944.6	1,096.4	1,275.0	1,912.6	325.0	2,028.4	1,674.4	724.8	19,326
SFP per requirement	6.28	10.40	8.91	6.71	7.59	5.91	6.28	6.46	8.05	6.88
LF/EP Ratio	16.22%	16.96%	21.79%	13.48%	18.10%	11.67%	19.76%	15.65%	22.03%	17.02%



Executing the Estimation Plan – Determining Hours

- The final SFP count was input into SEER-SEM.
 - Existing functions, not designed for reuse.
 - Preprogrammed factors, based on the knowledge bases selected, that assigned a level of reuse appropriate for each component.
- With the assumption that software strangler migration was being used and the fact that the requirements are stable and complete, a limited distribution of +/- 10% was applied.

Executing the Plan (cont)

- JHU APL assisted the cost team in establishing appropriate assumptions for the SEER-SEM model.

SEER Settings/Inputs by Component					
Component	Language(2)	Application(2)	Acquisition Method (5)	Development Method(5)	Software Phase at Estimate
Component 1	C#	Command/Control	Concept Reuse	Full Agile	Design
Component 2	SQL	Database App	Modification, Average	Full Agile	Code
Component 3	Javascript	Mission Planning & Analysis	Concept Reuse	Full Agile	Design
Component 4	C++	Communications	Concept Reuse	Full Agile	Design
Component 5	3rd Gen	Data Mining	Concept Reuse	Full Agile	Design
Component 6	3rd Gen	Database App	Concept Reuse	Full Agile	Design
Component 7	3rd Gen	Simulation	Concept Reuse	Full Agile	Design
Component 8	Visual Basic	Mission Planning & Analysis	Concept Reuse	Full Agile	Design
Component 9	3rd Gen	Mission Planning & Analysis	Modification, Average	Full Agile	Code

Assumptions:
1) All function points are pre-existing functions, not designed for reuse. Assumes some reduction in effort vs "green space" coding.
2) Assumed the same as previous estimate based on SLOC, verified with IT leads on 7/20/23.
3) Development Standard is IEEE full.
4) Due to move to microservices, may result in some functionality and design changes. Requirements are assumed stable.
5) APL provided assumptions as a result of their modernization study

Estimate Results

Quick Estimate		Person Hours by Labor Category	
Item	Estimate		
PROJECT: TMPC Modernization			
Development Schedule Months	107.37		
Development Effort Months	7,971.61		
Development Effort Hours	1,211,685		
Development Labor Cost	0		
Maintenance Schedule Months	0.00		
Maintenance Effort Months	0.00		
Maintenance Effort Hours	0		
Maintenance Labor Cost	0		
Delivered Defects	143		

All results shown are at the 50% confidence interval

Activity	Hrs
System Requirements design	17,048
Software Requirements analysis	50,032
Preliminary design	95,079
Detailed design	190,798
Coding and unit test	358,371
Component integration and test	178,858
Program Testing	24,226
System Integration through OT&E	297,272
	1,211,686

Component	Hours
Component 1	605,951
Component 2	4,130
Component 3	117,309
Component 4	72,715
Component 5	19,750
Component 6	119,967
Component 7	139,176
Component 8	127,258
Component 9	5,430
	1,211,686

Benefits to Program

- Independent assessment of the effort required to modernize TMPC software rather than leaning on vendor proposals.
- Defensible, rational, and repeatable given a solid underlying data set and methodology.
- Program can add and remove requirements from the estimate at the SSS level to assess impacts of requirements changes to the effort.
- The estimate is now an integrated tool for program decision making.

Path Forward

- TMPC is now also positioned to build a data set based on actual results.
- Should now be able to track delivered requirements and associate them to their SFP count and actual hours.
- Long-range goal is to build a data set that can be used going forward to either calibrate parametric models or develop their own data based Cost Estimating Relationships (CERs).

Conclusions and Lessons Learned

- Software Requirements Specifications are a better source of data for SFP counts than SSS.
- It would be optimal to have engineering staff complete the SFP count independent of the cost team.
- Estimators should admit what they do not know, seek expert inputs, and involve the entire team in the process as much as feasible.
- The SFP count took about four months to complete, purchasing a Commercial Off-The-Shelf (COTs) SFP automation counting software license might decrease time for this work.