From Zero to Hero: Roadmapping in an Agile World

Jacob Blackthorn and Kristen Marquette

Quantech Services, Inc.

Table of Contents:

I. Abstract

"We're Agile! We don't need to tell you what we're working on! It'll be done when it's done." This is a common adage in the Agile Community and has led to numerous failed audits. However, Agile Roadmapping IS possible! Our team has worked directly with Program IPTs to develop Roadmaps that incorporate projected work in an iterative process utilizing past performance data. We were then able to turn these Content Driven Roadmaps detailing the time-phased product backlog directly into an FTE requirement utilizing an Analysis of Agile Data taken directly from the Software Teams Jira tool. Join us to learn how our program went from zero or hero, as they are now considered the golden standard of agile estimating.

II. Problem Statement

Level of effort (LOE) estimation is primarily used in large agile DevSecOps programs with intensive software development. It is often used to capture a large number of software teams, primarily focusing on quantity of full-time-equivalents (FTEs). While this methodology is used across many agile software intensive programs, it does not provide the same level of credibility or linkage to requirements as compared to many methodologies used by non-software-based programs. LOE estimates can be difficult to defend, however their methodologies can be bolstered to provide higher fidelity in an estimate.

Agile programs are often challenged to develop methodologies with more fidelity and credibility. The methodology that this paper utilized is a content-driven roadmap (CDR). This CDR linked historical productivity data to support the development of a Cost Analysis Requirements Description (CARD) and other cost estimating activities. CDRs directly supported the program office in identifying the capability development timeline. They were also useful in identifying all planned capabilities, their sizing, complexity, and priority levels.

III. Requirements for Successful Analysis

To be successful with this analysis, there are many criteria that were necessary. First and foremost was program office buy-in. This undertaking would not have been possible without program office support. To build a success CDR, there was a significant amount of engineering and technical expertise required to project the proper level of complexity associated with future efforts. To facilitate technical conversations, regular meetings with the engineering and functional teams was essential, as it provided an opportunity to deep dive the data and correct for any abnormalities. Additionally, attending program events provided insight into software development planning and relevant programmatic events.

The second criteria was the standardization of Jira data that software developers utilize for sprint planning. To ensure the accuracy of the data analysis the data was regularly groomed, validating the required fields, which will be discussed further in Requirements Breakdown and Jira Analysis.

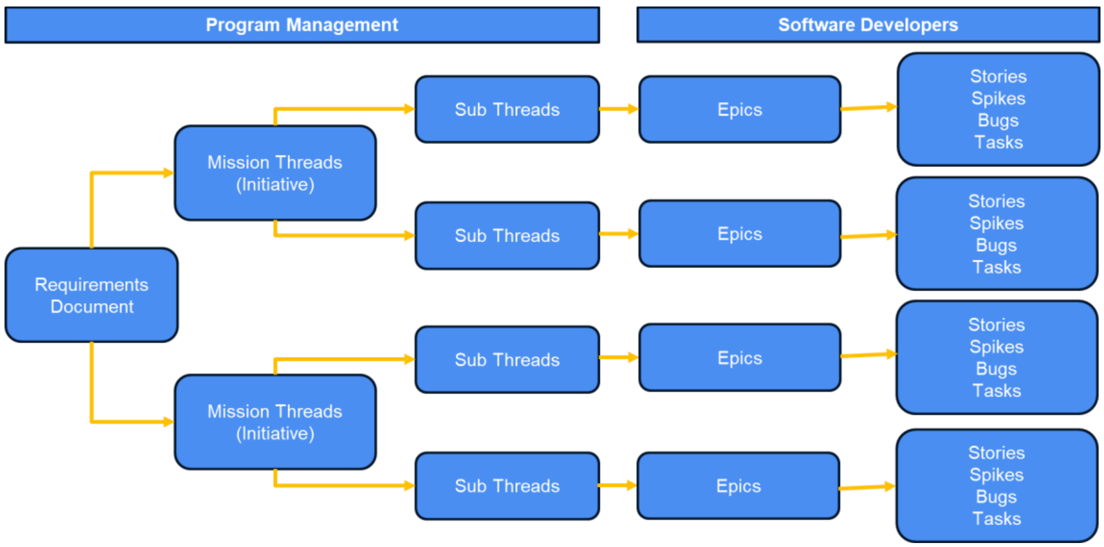
Lastly, it was necessary to have consistent reporting. The analysis required data collection of staffing and program management data on a monthly basis via programmatic management reports. On the technical side, after each Program Increment (PI), the Jira database would be captured.

IV. Requirements Traceability

To begin the process of creating and utilizing a CDR, requirements were properly traced from the ground floor software development work up to the requirements document by the program office. Essentially, the requirements document was translated into an exhaustive list of mission threads, or initiatives in Jira terminology. This list of mission threads was then broken into a number of sub threads that fully represented the scope of the capabilities to be developed.

At this point, software developers work to decompose each sub thread into a number of epics (chunks of work to be accomplished over a few PIs). In turn, these epics were also decomposed into the lower-level stories, bugs, spikes, or tasks. It is important to note that the total number of mission threads and sub threads did not change over time, while the total number of epics grew from PI to PI. This is due to the fact that software developers realized more nuance behind each sub thread as software development continued. A visual of this breakdown is shown in Graph 1. This entire hierarchy was built directly into Jira, which supported data traceability in the model.

Graph 1: Requirement Traceability



The subsequent sections will walk through the analysis using fabricated data.

V. Jira Analysis

a. Jira Tagging

For this analysis to be successful, the Jira database required rigorous management to ensure certain fields were routinely updated. A sample data export is shown is below.

Graph 2: Jira Data Pull

Issue key	Summary	Issue Type	Status	Resolution	Created	Resolved	Epic Link	Parent Link	Story Points	Dev Team
ISSUE-001	Core Mission Thread tying to Requirements	Initiative	In Progress		3/19/2024					
ISSUE-002	Epic work that ties to Mission Thread	Epic	Done		3/19/2024			ISSUE-001		Team 1
ISSUE-003	Epic work that ties to Mission Thread	Epic	In Progress		4/15/2024			ISSUE-001		Team 2
ISSUE-004	Epic work that ties to Mission Thread	Epic	In Progress		5/8/2024			ISSUE-001		Team 3
ISSUE-005	Epic work that ties to Mission Thread	Epic	In Progress		9/9/2024			ISSUE-001		Team 4
ISSUE-006	Epic work that ties to Mission Thread	Epic	To Do		10/29/2024			ISSUE-001		Team 5
ISSUE-007	Story that ties to Epic	Story	Done	Done	3/19/2024	3/19/2024	ISSUE-002		3	
ISSUE-008	Story that ties to Epic	Story	Done	Done	3/19/2024	3/19/2024	ISSUE-002		5	
ISSUE-009	Story that ties to Epic	Story	Done	Done	4/15/2024	4/15/2024	ISSUE-003		1	
ISSUE-010	Story that ties to Epic	Story	Done	Done	4/15/2024	4/15/2024	ISSUE-003		8	
ISSUE-011	Story that ties to Epic	Story	To Do		4/15/2024		ISSUE-003			
ISSUE-012	Story that ties to Epic	Story	Done	Done	5/8/2024	5/8/2024	ISSUE-004		13	
ISSUE-013	Story that ties to Epic	Story	To Do		5/8/2024		ISSUE-004			
ISSUE-014	Story that ties to Epic	Story	Done	Done	9/9/2024	9/9/2024	ISSUE-005		5	
ISSUE-015	Story that ties to Epic	Story	To Do		9/9/2024		ISSUE-005			
ISSUE-016	Story that ties to Epic	Story	To Do		10/29/2024		ISSUE-006			

The standard fields required for this analysis were **Issue Key**, **Summary**, and **Created** (Issue Creation Date).

Issue Type was necessary to show which level of the hierarchy any issue resides in.

Status represents what milestone the issue has reached (To Do, In Progress, and Done are the minimal required).

Resolution and **Resolved** (Resolution Date) were important to determine when a given story, spike, bug, or task has been completed.

Epic Link was important to show which epic any story, spike, or task was linked to. This then allowed the team to roll the stories up to their parent epics to determine how much of the epic was complete at any given point.

Likewise, **Parent Link**, or Initiative Link, allowed the team to see what mission thread or initiative any epic was linked to.

Story Points were necessary for the lowest level elements (story, spike, bug, or task) to show the rough level of effort needed for that element. As a note, Story Points are not rigorously defined but should be consistent within the team working them.

Dev Team was a necessary field and should be minimally labeled by epic, where each epic was worked on by only one team in a given PI.

b. Mapping Hours Burned to Epics Worked

Since traditionally, most contractors or organizations do not track hours burned by Jira, the team was unable to collect a perfect metric of the number of hours burned by epic. However, as hours burned were aggregated by individual or team, the team was able to approximate the number of hours burned per epic by utilizing story points as a weighting factor.

Graph 3: Mapping Hours to Completed Epics

Hours Burned	Aug-23	Sep-23	Oct-23	Nov-23	Dec-23	Jan-24	Feb-24	Mar-24	Apr-24	May-24	Jun-24	Jul-24	Aug-24	Sep-24	Oct-24
Software Team		PI - 5			PI - 6			PI - 7			PI - 8			PI - 9	
Team 1	1280	1280	1440	1280	1280	1120	960	1120	1280	1440	1440	1280	1280	1280	1280
Team 2	1440	1440	1440	1600	1760	1920	1760	1600	1600	1600	1760	1440	1440	1280	1280
Team 3	1440	1600	1440	1280	1280	1600	1440	1600	1440	1600	1600	1440	1600	1280	1280
Team 4	1760	1600	1760	1600	1440	1600	1440	1600	1440	1440	1600	1280	1440	1600	1760
Team 5	1760	1600	1440	1280	1280	1280	1440	1280	1120	1120	1280	1440	1440	1440	1600

			Story Points Resolved					Appx Hours				
Epic Key	Sub-Thread / Jira Epic	PI - 6	PI - 5	PI - 6	PI - 7	PI - 8	PI - 9	PI - 5	PI - 6	PI - 7	PI - 8	PI - 9
ISSUE-100	Software Team 2 - Epic 1	Team 2	7	8	0	0	0	643	1564	0	0	0
ISSUE-101	Software Team 2 - Epic 2	Team 2	3	0	2	2	4	276	0	472	356	222
ISSUE-102	Software Team 2 - Epic 3	Team 2	9	0	0	12	6	827	0	0	2133	333
ISSUE-103	Software Team 2 - Epic 4	Team 2	0	0	0	12	5	0	0	0	2133	278
ISSUE-104	Software Team 2 - Epic 5	Team 2	0	0	1	1	11	0	0	236	178	611
ISSUE-105	Software Team 2 - Epic 6	Team 2	13	0	0	0	6	1195	0	0	0	333
ISSUE-106	Software Team 2 - Epic 7	Team 2	0	10	0	0	13	0	1956	0	0	722
ISSUE-107	Software Team 2 - Epic 8	Team 2	0	0	13	0	5	0	0	3070	0	278
ISSUE-108	Software Team 2 - Epic 9	Team 2	7	4	0	0	11	643	782	0	0	611
ISSUE-109	Software Team 2 - Epic 10	Team 2	8	5	0	0	9	735	978	0	0	500
ISSUE-110	Software Team 2 - Epic 11	Team 2	0	0	5	0	2	0	0	1181	0	111

The table at the top of Graph 3 shows a breakout of five teams and the number of hours each team burned each month. For this example, one PI equates to a three-month period. The table at the bottom of graph 3 lists out epics worked by **Team 2** and the number of story points resolved in each epic broken out by PI. To approximate the number of hours burned by each epic, this methodology used story points as a weighting factor against hours in that PI. For example, in **P1-6** (November 2023 through January 2024), **Team 2** burned **5,280** hours.

$$1{,}600\ hrs + 1{,}760\ hrs + 1{,}920\ hrs = 5{,}280\ hrs$$

Additionally, **Team 2** burned a total of **27** story points over the course of **PI-6** over a number of Epics.

$$8 + 10 + 4 + 5 = 27$$

The **5,280** hours were then divided proportionally towards epics based on the number of story points they completed compared to the overall. For example, **Issue-100** had **8** story points completed in **PI-6**, so **8 divided by 27** was the factor applied to the **5,280** hours to yield an approximate burn of **1,564** hours on Issue-100 in Program Increment 6.

$$\frac{8\ story\ points}{27\ story\ points} * 5{,}280\ hrs = 1{,}564\ hrs$$

This methodology was applied to every epic in every PI based on the teams working those epics. This was to make sure that story points were not mixed between different teams who may have different story point methodologies, as well as to keep the team's hours burned to only the work they had accomplished.

c. Data Point Selection Process

Now that this paper has shown how the team tracked approximate effort hours for Jira epics, it's important to show how the data was used as an analogy for future estimates. In order for an epic to be a valid analogy to compare to future epics, the full scope of the epic needed to be captured. Therefore, each valid analogy data point had to fulfill three criteria.

Criteria 1: All work on an epic needed to be captured within the scope of the data collection, so only epics that began after data collection began are viable. For example, in Graph 4, Issue-100 and Issue-101 were already "In Progress" below at the first data collection in PI-5. Therefore, they did not satisfy the first criteria.

Criteria 2: An epic had to reach a completion state within the data collection, so only epics that had reached "Done" or similar status are viable. For example, Issue-100 through Issue-106 had reached "Done" status within the data collection set.

Criteria 3: An epic needed Story Points assigned to it at the lower-level issues (stories, spikes, bugs, or tasks). If there were no story points assigned to the epic, there would be no approximate hours worked for that epic. This could be a data maintenance issue, as teams should be story pointing all of their lower-level issues. However, it is possible, that a given epic completes with zero story points if it is a trivial amount of effort, or the epic was rejected.

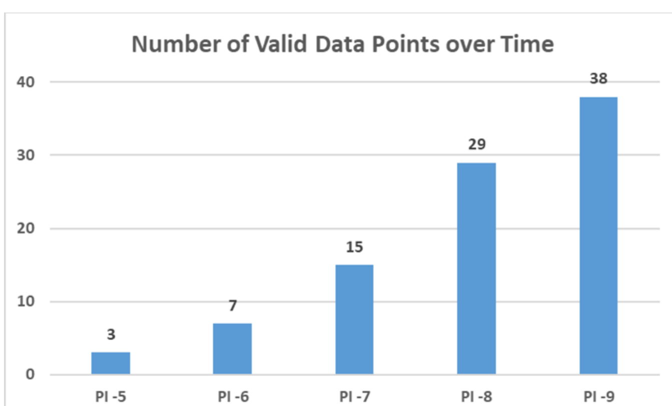
To re-iterate, valid analogy data points had to fulfill all three criteria, so while the example of Issue-100 may have satisfied Criteria 2 and 3, it did not satisfy Criteria 1. Therefore, Issue-100 was **not** considered a valid data point.

Graph 4: Data Point Criteria

Epic Key	PI -5	PI -6	PI -7	PI -8	PI -9
ISSUE-100	In Progress	Done	Done	Done	Done
ISSUE-101	In Progress	In Progress	Done	Done	Done
ISSUE-102	To Do	In Progress	Done	Done	Done
ISSUE-103	To Do	To Do	In Progress	Done	Done
ISSUE-104		To Do	In Progress	In Progress	Done
ISSUE-105		To Do	In Progress	In Progress	Done
ISSUE-106			To Do	In Progress	Done
ISSUE-107			To Do	In Progress	In Progress
ISSUE-108			To Do	In Progress	In Progress
ISSUE-109				To Do	In Progress
ISSUE-110				To Do	In Progress

As data collection continued in every PI, the number of valid data points used in the analysis over time grow. Graph 5 shows that at the end of PI-5, only three epics were valid data points, while at the end of PI-9, the set had increased to 38.

Graph 5: Valid Data Points Over Time



d. Stratified Data Point Results

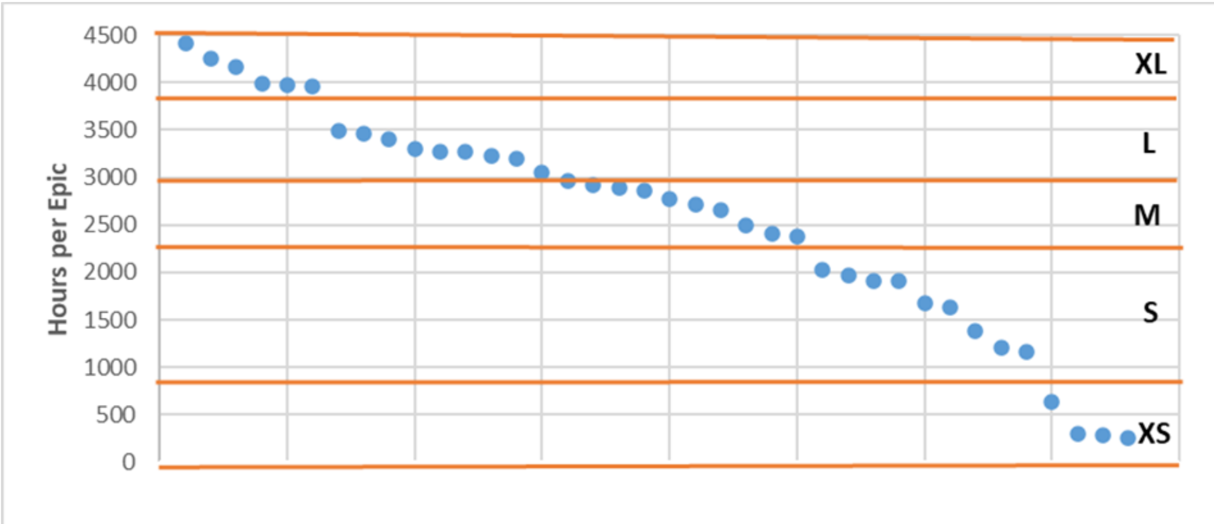
Once the data points are defined, they needed to be stratified into five bins or T-shirt sizes for easy comparisons and analysis. Graph 6 shows a breakdown of the 38 data points or epics that met the three criteria in the above section. These data points are bucketed into a T-shirt size using K-Means, which will be discussed further in the next section. This data was then used as reference data for the estimation and quantification of the CDR.

Graph 6: Stratified Data Point Results

	Jira Data as of PI 9		Total (PI 5-9)		Bin Size
Epic Key	Sub-Thread / Jira Epic	Component	SP Cmp.	Dev Hrs.	
ISSUE-331	Epic Work Name 31	Team 1	441	4410	XL
ISSUE-322	Epic Work Name 22	Team 2	426	4260	XL
ISSUE-327	Epic Work Name 27	Team 2	417	4170	XL
ISSUE-323	Epic Work Name 23	Team 3	443	3987	XL
ISSUE-316	Epic Work Name 16	Team 1	441	3969	XL
ISSUE-336	Epic Work Name 36	Team 1	396	3960	XL
ISSUE-330	Epic Work Name 30	Team 5	437	3496	L
ISSUE-338	Epic Work Name 38	Team 3	347	3470	L
ISSUE-306	Epic Work Name 6	Team 1	425	3400	L
ISSUE-325	Epic Work Name 25	Team 5	412	3296	L
ISSUE-310	Epic Work Name 10	Team 5	364	3276	L
ISSUE-334	Epic Work Name 34	Team 4	327	3270	L
ISSUE-314	Epic Work Name 14	Team 4	359	3231	L
ISSUE-311	Epic Work Name 11	Team 1	356	3204	L
ISSUE-319	Epic Work Name 19	Team 4	339	3051	L
ISSUE-305	Epic Work Name 5	Team 5	370	2960	M
ISSUE-317	Epic Work Name 17	Team 2	366	2928	M
ISSUE-304	Epic Work Name 4	Team 4	289	2890	M
ISSUE-332	Epic Work Name 32	Team 2	357	2856	M
ISSUE-335	Epic Work Name 35	Team 5	347	2776	M
ISSUE-301	Epic Work Name 1	Team 1	301	2709	M
ISSUE-313	Epic Work Name 13	Team 3	266	2660	M
ISSUE-329	Epic Work Name 29	Team 4	249	2490	M
ISSUE-337	Epic Work Name 37	Team 2	301	2408	M
ISSUE-324	Epic Work Name 24	Team 4	264	2376	M
ISSUE-312	Epic Work Name 12	Team 2	203	2030	S
ISSUE-326	Epic Work Name 26	Team 1	218	1962	S
ISSUE-318	Epic Work Name 18	Team 3	239	1912	S
ISSUE-328	Epic Work Name 28	Team 3	238	1904	S
ISSUE-302	Epic Work Name 2	Team 2	168	1680	S
ISSUE-308	Epic Work Name 8	Team 3	181	1629	S
ISSUE-320	Epic Work Name 20	Team 5	172	1376	S
ISSUE-315	Epic Work Name 15	Team 5	134	1206	S
ISSUE-307	Epic Work Name 7	Team 2	146	1168	S
ISSUE-303	Epic Work Name 3	Team 3	80	640	S
ISSUE-309	Epic Work Name 9	Team 4	30	300	XS
ISSUE-321	Epic Work Name 21	Team 1	35	280	XS
ISSUE-333	Epic Work Name 33	Team 3	26	260	XS

A further visualization of the bin collections is seen below in **Graph 7**. The clusters of data binned into XS through XL buckets were a result of the K-Means analysis.

Graph 7: Visual of Stratified Data Points



e. K-Means Clustering

K-Means is a statistical technique used for vector quantization (Piech, 2013). K-Means is a clustering algorithm that finds the best centroids for the data by assigning data points to clusters based on the centroids and by choosing centroids based on data points to cluster (Piech, 2013). This algorithm is shown below in Graph 9; it can also be calculated in Python or R software.

Graph 8: K-Means Algorithm

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {

   For every i, set

   $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

   For each j, set

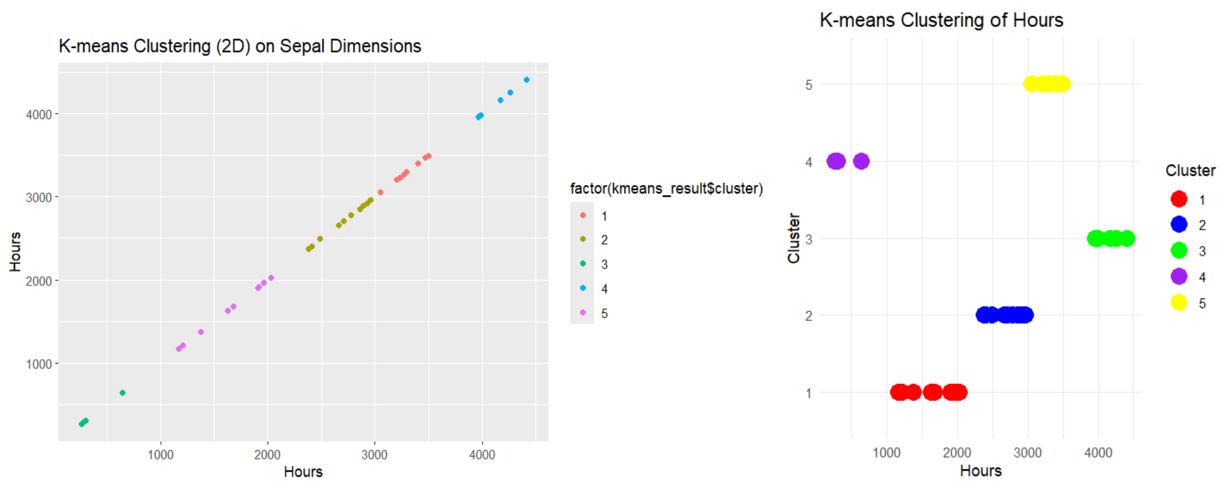
   $$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

   }

Source: Piech, 2013

K-Means was leveraged against the data set shown in Graph 6 analyzing the different hours worked per epic. K-Means Clustering will divide observations into clusters based on the mean of the cluster or centroid closest to the data point (Piech, 2013). It took the above data points and assigned them to the correct bins, portrayed in the Graph 9, showing the data binned into five buckets, representing T-shirt sizes XS through XL.

Graph 9: K-Means Clustering Graphics



f. Data Point Set Comparison

While this analysis provides a credible, high-fidelity estimate, data collection and normalization take time. As shown below, in Graph 10 the number of valid data points grow each PI. To gauge the normalization of data over time, the data bin categories were compared over multiple PIs. Graph 10 provides a snapshot of data bin collections for PI-8 and PI-9 and their relative change.

Graph 10: Data Point Comparison

Analogous Data Bins						
Bin Size	PI - 8		PI - 9			Change
	Avg Hrs	# Data Pts	Avg Hrs	# Data Pts	Std Dev	
XS	265	2	280	3	20	6%
S	1424	6	1551	10	447	9%
M	2560	9	2705	11	216	6%
L	3450	8	3300	9	139	-4%
XL	4579	4	4126	5	186	-10%
	Total	29	Total	38	`	

In the previous PIs (PI-8 and prior), the data set was very limited, with only 29 epics being classified as viable data points. Therefore, the team utilized a simple box-and-whisker plot to normalize the data into five T-shirt sizes. By PI-9, the data set had grown to 38, which allowed more robust K-Means clustering analysis, as shown above, to properly allocate epics into five distinct bin sizes with limited change in average hours for the clusters. As data continues to accrue each year, the team expects these data bins to solidify into more concrete sets.

VI. The Content-Driven Roadmap

a. Creation of the Content-Driven Roadmap

Utilizing the breakdown of official requirements into mission threads, sub threads, and epics, the team created a tabular roadmap to project work to go, as shown in Graph 11.

Graph 11: Content-Driven Roadmap

Content-Driven Roadmap	Backlog as of PI - 9				FY25	FY26	FY27	FY28
Mission Thread/Sub-Thread/Epic	Jira Key/ID	Status	Team	Phase Filter	Bin	Bin	Bin	Bin
Mission Thread 1				Mission Thread				
Sub-thread 1				Sub-Thread				
Epic 1	ISSUE-401	Done	Team 1	Jira Epic				
Epic 1				Critical Enhancements				
Epic 1				Product Support	XS	XS	XS	XS
Epic 2	ISSUE-402	To Do	Team 1	Jira Epic				
Epic 2				Critical Enhancements		M		
Epic 2				Product Support			XS	XS
Epic 3	ISSUE-403	In Progress	Team 1	Jira Epic				
Epic 3				Critical Enhancements	M	M		
Epic 3				Product Support			XS	XS
Sub-thread 2				Sub-Thread				
Epic 1	ISSUE-404	In Progress	Team 2	Jira Epic				
Epic 1				Critical Enhancements	XL	L		
Epic 1				Product Support			M	M
Epic 2	ISSUE-405	In Progress	Team 2	Jira Epic				
Epic 2				Critical Enhancements	M	M		
Epic 2				Product Support			S	S
Sub-thread 3				Sub-Thread				
Epic 1	ISSUE-406	In Progress	Team 3	Jira Epic				
Epic 1				Critical Enhancements	L	L	L	
Epic 1				Product Support				M
Epic 2	ISSUE-407	To Do	Team 3	Jira Epic				
Epic 2				Critical Enhancements			L	XL
Epic 2				Product Support				

This work was stratified into two categories: Critical Enhancements, which represented new capability or new development work, and Product Support, which represented the usual maintenance, bug fixes, and general work necessary to keep the capabilities functioning. In conjunction with the technical teams as mentioned in Requirements for Successful Analysis, epics were sized by a relative level complexity of work over the next four years. The four-year projection was to try and balance the traditional short-term estimating view of agile software development with the usual 7-year outlook of the Fiscal Year Defense Program (FYDP) Process. The relative bucket sizes that each area fell into are compared to regular T-shirt sizes (XS, S, M, L, XL). These sizes were directly pulled from the historical data in Graph 6 so the technical teams could easily compare future work to completed work.

b. Jira Analogy Results

Ultimately, the T-shirt sized roadmap was directly converted into hours, utilizing the average hours of each T-shirt size, shown in Graph 12. The average effort hours associated with T-shirt size, XS, S, M, L, XL were 280, 1,551, 2,705, 3,299, and 4,126 respectively. These hours were then easily converted into an FTE count based on an assumption of 1 FTE is equivalent to 1920 hours per year. The sizes in the CDR were transformed into FTE, as shown below in Graph 12.

Graph 12: Results

Analogous Data Bins - K Means Clustering			
Bin Size	Avg Hrs	Low Hrs	High Hrs
XL	4126	3960	4410
L	3299	3051	3496
M	2705	2376	2960
S	1551	640	2030
XS	280	260	300

Roadmap Totals	Epic Counts				Hours				FTE			
Phase	FY25	FY26	FY27	FY28	FY25	FY26	FY27	FY28	FY25	FY26	FY27	FY28
Critical Enhancements	57	54	38	43	76,945	75,266	53,262	68,208	40.1	39.2	27.7	35.5
Product Support	63	73	97	104	47,440	56,594	66,449	68,409	24.7	29.5	34.6	35.6
Total	**120**	**127**	**135**	**147**	**124,386**	**131,859**	**119,711**	**136,617**	**64.8**	**68.7**	**62.3**	**71.2**

c. CDR Scaling Factors to Account for Unknowns

As data analysis of Jira epics only considered epics that were completed in the Pre-Integration and Test (Pre-I&T) Software Development Phase, and not necessarily the Integration and Test (I&T) Phase, additional scaling factors were necessary to fully reach the necessary number of FTE required to complete the work. This section will detail the capability roadmap scaling factors used to correct human error of roadmap T-shirt sizing and ensure the full requirement was captured. These scaling factors include I&T Software Development Phase, New Scope/New Requirements, and Sizing Accuracy.

The first piece was already estimated, the Pre-I&T Software Development Phase, which was already captured in the above roadmap results. These results are also shown in Graph 13.

Graph 13: Roadmap Results – Pre-I&T Software Development Phase

Roadmap Totals	FTE			
Phase	**FY25**	**FY26**	**FY27**	**FY28**
Critical Enhancements	40.1	39.2	27.7	35.5
Product Support	24.7	29.5	34.6	35.6
Total	**64.8**	**68.7**	**62.3**	**71.2**

SW Development Total					Factor
Core SW Development					
Roadmap SW Dev (Pre-I&T)	64.8	68.7	62.3	71.2	Roadmap
SW Dev (I&T Phase)					
SW Dev (New Scope/Req Growth)					
SW Dev (Sizing Factor)					

The first scaling factor, associated with the I&T Software Development Phase, was captured by comparing the hours worked per epic on the Pre-I&T Software Development Phase with hours worked within the I&T Software Development Phase. Graph 14 shows the hours completed per Epic for Pre-I&T and during I&T. The total hours worked in Pre-I&T were 125,000, while hours worked in I&T were 49,875, which means 39.3 percent of the software development associated with Integration and Test was not captured in the roadmap.

$$\frac{49875\ (I\&T\ Hours)}{125000\ (Pre-I\&T\ Hrs)} * 100\ percent = 39.3\ percent$$

Therefore, the analysis added a 39.3 percent scaling factor. As further I&T data is accrued this scaling factor will become redundant as I&T will be captured in the roadmap directly. Both software development phases together account for the Core SW Development FTE for estimating purposes.

Graph 14: CDR Scaling Factor – I&T Software Development Phase

FY24 Actuals				
Epics	Status	Hours Completed		% Add'l Work
		Pre-I&T Phase	I&T Phase	
Epic 1	Done	2705	280	10%
Epic 2	Done	3299	80	2%
Epic 3	Done	4126	2705	66%
Epic 4	Done	1551	500	32%
Epic 5	Done	2705	1000	37%
Epic 6	Done	1551	300	19%
Epic 7	Done	4126	2500	61%
Epic 8	Done	2705	1000	37%
Epic 9	Done	4126	2200	53%
Epic 10	Done	1551	950	61%
Epic 11	Done	280	90	32%
...	....	...	...	...
Total		**125000**	**49875**	**39.3%**

SW Development Total					Factor
Core SW Development	90.2	95.7	86.9	99.1	
Roadmap SW Dev (Pre-I&T)	64.8	68.7	62.3	71.2	Roadmap
SW Dev (I&T Phase)	25.5	27.0	24.5	28.0	39.3%
SW Dev (New Scope/Req Growth)					
SW Dev (Sizing Factor)					

The next scaling factor was necessary to capture the additional work associated with new scope or requirements growth. In other words, this was meant to capture any new requirements or scope that may have been added to the portfolio over time that was inherently not captured by a CDR based on current requirements.

The methodology used for this factor was to compare the epics that were captured in the 2023 CDR to the epics that were added after the roadmap was created. This is captured in Graph 15, showing the numerous new scope epics that were worked, but not captured in the 2023 CDR.

Graph 15: CDR Scaling Factor – New Scope / Requirements Growth

2023 Content Driven Roadmap		
Epics	**Status**	**Hours Burned**
Epic 1	To Do	288
Epic 2	To Do	400
Epic 3	To Do	650
Epic 4	To Do	120
Epic 5	To Do	160
Epic 6	To Do	8
Epic 7	To Do	20
Epic 8	To Do	0
Epic 9	To Do	6
Epic 10	To Do	120
Epic 11	To Do	55

New Scope Epics (Not included in 2023 Roadmap)		
Epics	**Status**	**Hours Burned (FY24)**
Epic 12	To Do	460
Epic 13	To Do	180
Epic 14	To Do	840
Epic 15	To Do	12
Epic 16	To Do	60
Epic 17	To Do	420
Epic 18	To Do	620
…	….	…
Total		**16452**

The hours dedicated to new scope requirements was 16,452 hours, which is equivalent to about 8.6 FTE, assuming one FTE is equivalent to 1,920 hours. The 2023 CDR captured 90.2 FTE, which means the work towards new scope or new requirements was about 9.5 percent.

$$\frac{16452\ (Hours)}{1920\ (HPY)} = 8.56\ FTE$$

The hours dedicated to new scope requirements was 16,452 hours, which is equivalent to about 8.6 FTE.

$$\frac{8.56\ (FTE)}{90.2\ (Total\ FTE)} * 100\ percent = 9.5\ percent$$

The 2023 CDR captured 90.2 FTE for Core SW Development, which means the work towards new scope or new requirements was about 9.5 percent. As more data is accrued, this factor will be refined and become more exact.

Graph 16: CDR Scaling Factor – New Scope/Req Growth Results

Hours Burned Towards New Scope Epics (Epics not in 2023 Roadmap) PI 5-9			
Hours	FTE	Core SW Dev FTE	% Add'l
16452	8.6	90.2	**9.5%**

SW Development Total					Factor
Core SW Development	90.2	95.7	86.9	99.1	
Roadmap SW Dev (Pre-I&T)	64.8	68.7	62.3	71.2	Roadmap
SW Dev (I&T Phase)	25.5	27.0	24.5	28.0	39.3%
SW Dev (New Scope/Req Growth)	8.6	9.1	8.3	9.4	9.5%
SW Dev (Sizing Factor)					

The final scaling factor was necessary to adjust the accuracy of the T-shirt sizing by the technical team. To do this, the previous year's CDR was analyzed against actuals. As shown in Graph 17, the 2023 CDR provided a forecast of approximate hours worked per epic based on T-shirt sizing in the roadmap. These forecasted hours were compared to actual hours on completed epics in FY24.

Graph 17: CDR Scaling Factor – Sizing Factor

2023 Content Driven Roadmap				FY24 Actuals			
Epics	Status	Roadmap Sizing	Forecast Hrs (FY24)	Epics	Status	Roadmap Sizing	Actual Hrs
Epic 1	To Do	XS	280	Epic 1	Done	M	2705
Epic 2	To Do	S	1551	Epic 2	Done	L	3299
Epic 3	To Do	M	2705	Epic 3	Done	XL	4126
Epic 4	To Do	L	3299	Epic 4	In Progress	S	1551
Epic 5	To Do	M	2705	Epic 5	In Progress	M	2705
Epic 6	To Do	XS	280	Epic 6	Done	S	1551
Epic 7	To Do	XL	4126	Epic 7	Done	XL	4126
Epic 8	To Do	S	1551	Epic 8	Done	M	2705
Epic 9	To Do	L	3299	Epic 9	In Progress	XL	4126
Epic 10	To Do	XL	4126	Epic 10	In Progress	S	1551
Epic 11	To Do	XS	280	Epic 11	In Progress	XS	280
…	….	…	…	…	….	…	…
Total			**20426**	Total			**24327**

While the roadmap estimated an approximate 20,426 hours, actuals hours worked in FY24 totaled 24,327, with a delta of 3,901 hours. This means that 19.1 percent of the software development work is not currently estimated in the roadmap due to sizing constraints.

$$\left(\left(\frac{24327}{20426}\right) - 1\right) * 100\ percent = 19.1\ percent$$

Therefore, this analysis added a 19.1 percent scaling factor. As more data is accrued, this factor will be refined and become more exact.

Graph 18: CDR Scaling Factor – Sizing Factor Results

Projected (2023 Roadmap) vs. Actual Completed Epic SW Dev Hrs burned in FY24		
Forecast Hrs	**Actual Hrs**	**% Add'l**
20426.0	24327.0	**19.1%**

SW Development Total	**116.1**	**123.0**	**111.7**	**127.5**	**Factor**
Core SW Development	90.2	95.7	86.9	99.1	
Roadmap SW Dev (Pre-I&T)	64.8	68.7	62.3	71.2	Roadmap
SW Dev (I&T Phase)	25.5	27.0	24.5	28.0	39.3%
SW Dev (New Scope/Req Growth)	8.6	9.1	8.3	9.4	9.5%
SW Dev (Sizing Factor)	17.2	18.3	16.6	18.9	19.1%

All three of these scaling factors together added higher fidelity to the estimate, accounting for human error and unforeseen future needs and therefore better capturing the scope of requirements.

d. CDR FTE Results

For explanation purposes, these factors were shown at the top level, but in order to show a proper representation of the FTE required at each individual requirement or mission thread level, the FTE were scaled appropriately at each mission thread and sub thread level. This provided a visual representation of the FTE necessary by each requirement, which served as a basis for schedule and resource allocation by the program office.

Graph 19: CDR FTE Results (w/ Scaling)

Mission Threads	FTE by Mission Thread						
	FY24	FY25	FY26	FY27	FY28	FY29	FY30
Mission Thread 1	**34.0**	**34.0**	**25.0**	**34.0**	**34.0**	**34.0**	**34.0**
Subthread 1	9.0	9.0	6.0	9.0	9.0	9.0	9.0
Subthread 2	10.0	10.0	8.0	10.0	10.0	10.0	10.0
Subthread 3	8.0	8.0	5.0	8.0	8.0	8.0	8.0
Subthread 4	7.0	7.0	6.0	7.0	7.0	7.0	7.0
Mission Thread 2	**15.0**	**15.0**	**14.0**	**15.0**	**15.0**	**15.0**	**15.0**
Subthread 1	6.0	6.0	5.0	6.0	6.0	6.0	6.0
Subthread 2	5.0	5.0	5.0	5.0	5.0	5.0	5.0
Subthread 3	4.0	4.0	4.0	4.0	4.0	4.0	4.0
Mission Thread 3	**9.0**	**9.0**	**8.0**	**9.0**	**9.0**	**9.0**	**9.0**
Subthread 1	3.0	3.0	3.0	3.0	3.0	3.0	3.0
Subthread 2	6.0	6.0	5.0	6.0	6.0	6.0	6.0
Mission Thread 4	**43.4**	**41.2**	**41.0**	**42.0**	**42.0**	**42.0**	**42.0**
Subthread 1	9.0	8.0	8.0	8.0	8.0	8.0	8.0
Subthread 2	10.0	9.0	9.0	9.0	9.0	9.0	9.0
Subthread 3	14.0	12.2	12.0	13.0	13.0	13.0	13.0
Subthread 4	4.0	4.0	4.0	4.0	4.0	4.0	4.0
Subthread 5	6.4	8.0	8.0	8.0	8.0	8.0	8.0
Mission Thread 5	**6.0**	**7.0**	**7.0**	**7.0**	**7.0**	**7.0**	**7.0**
Subthread 1	3.0	4.0	4.0	4.0	4.0	4.0	4.0
Subthread 2	3.0	3.0	3.0	3.0	3.0	3.0	3.0
Mission Thread 6	**12.0**	**12.0**	**12.0**	**11.9**	**11.9**	**11.9**	**11.9**
Subthread 1	6.0	6.0	6.0	6.0	6.0	6.0	6.0
Subthread 2	6.0	6.0	6.0	5.9	5.9	5.9	5.9
Mission Thread 7	**-**	**5.0**	**5.0**	**5.0**	**5.0**	**5.0**	**5.0**
Subthread 1	-	1.0	1.0	1.0	1.0	1.0	1.0
Subthread 2	-	4.0	4.0	4.0	4.0	4.0	4.0
Mission Thread 8	**-**	**3.4**	**2.9**	**7.2**	**7.2**	**7.2**	**7.2**
Subthread 1	-	2.0	2.0	2.0	2.0	2.0	2.0
Subthread 2	-	1.4	0.9	-	-	-	-
Subthread 3	-	-	-	4.0	4.0	4.0	4.0
Subthread 4	-	-	-	1.2	1.2	1.2	1.2
Total Roadmap FTE	**119.4**	**126.6**	**114.9**	**131.1**	**131.1**	**131.1**	**131.1**

VII. Conclusions

With immense support from the program office, the team was able to develop a content-driven roadmap directly tying all program requirements to a direct timeline with appropriate complexity sizing. Due to the rigorous maintenance of the Jira database, the team utilized detailed Jira analysis to quantify the roadmap to develop a rigorous and high-fidelity estimate. Additionally, the modular and granular functionality of this roadmap supported numerous drills and exercises, resulting in quick-turn responses to the program office on a variety of taskers and funding "What if" drills. Ultimately, this herculean effort resulted in a rigorous and detailed CARD and methodologies that have been called "The Golden Standard of Cost Estimation."

References

Piech, C. (2013). K Means. Stanford University. https://stanford.edu/~cpiech/cs221/handouts/kmeans.html