

RESEARCH REVIEW 2022

**Carnegie
Mellon
University**
Software
Engineering
Institute

ACE/PoPs

Program Information at the Speed of Relevance

NOVEMBER 14–16, 2022

William Nichols
Principal Investigator

Document Markings



Copyright 2022 Carnegie Mellon University.

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.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

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.

[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.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon® and CERT® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM22-0880

Automated Continuous Estimation for Continuous Deployment and Pipelines of Pipelines

Automation drives continuous integration and delivery of software but outpaces program control.

To solve this problem, automate data collection Model DSO systems with **Monte Carlo** and provide continuous reporting.

- Determine status.
- Project future events.
- Provide evidence for corrective actions.

Goal: Programs using DSO(DevSecOps) have constant access to information needed to monitor and control schedule and cost commitments.

Status and projection models should be available in real time.

Model and simulate pipeline-of-pipeline systems.

Automate data collection and Program Management Status Reporting for DevSecOps pipelines.

Directly collect data from DevSecOps pipeline tools

- Automate data collection, storage, and reporting
- Correlate data to project outcomes
- Present completion to-date and milestone predictions to Program Management in smart dashboards

Agenda

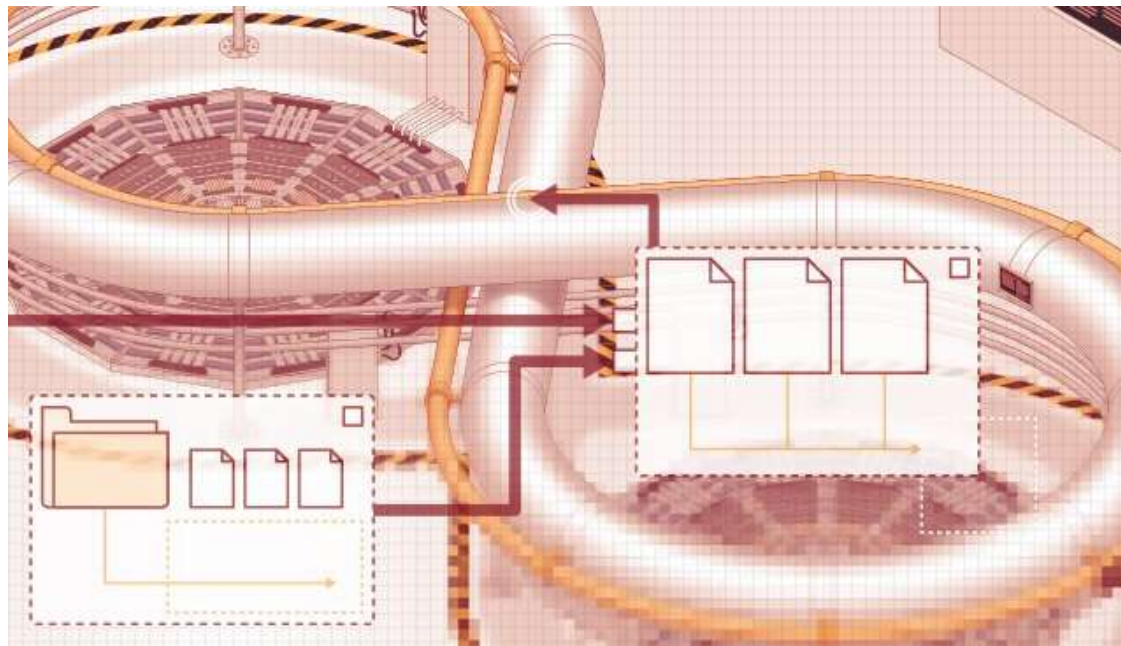
The programmatic challenge
with DevSecOps

What we did

- **Automating data collection and analysis**
- **Description of our prototype**
- **Scaling up to multiple pipelines**

Lessons learned

Conclusion



Program Managers Need Answers

As the PM of a complex agile program, you need to have an answer to these types of questions:

- If we drop everything to fix a high-priority vulnerability, when will my next two capabilities be delivered?
- A capability has slipped twice—when will it **really** be done and delivered?
- How many more teams would it take to deploy three specific capabilities in 6, 12, and 18 months?

As a Program Manager, What Do You Need to Know? And When Do You Need to Know It?

Where are we? (Capability % Complete, Schedule Consumed)

Where did we expect to be? (Planned Capability Complete and Schedule Variance Estimation accuracy)

Do we have enough time and money to get to the finish?

- When will we be done? (Schedule projections to complete)
- What will it cost? (Often dominated by Schedule and staffing)
- When *could* we have it? (“what-if” projections)
 - If we adjust priorities?
 - If we add staff or other resources?

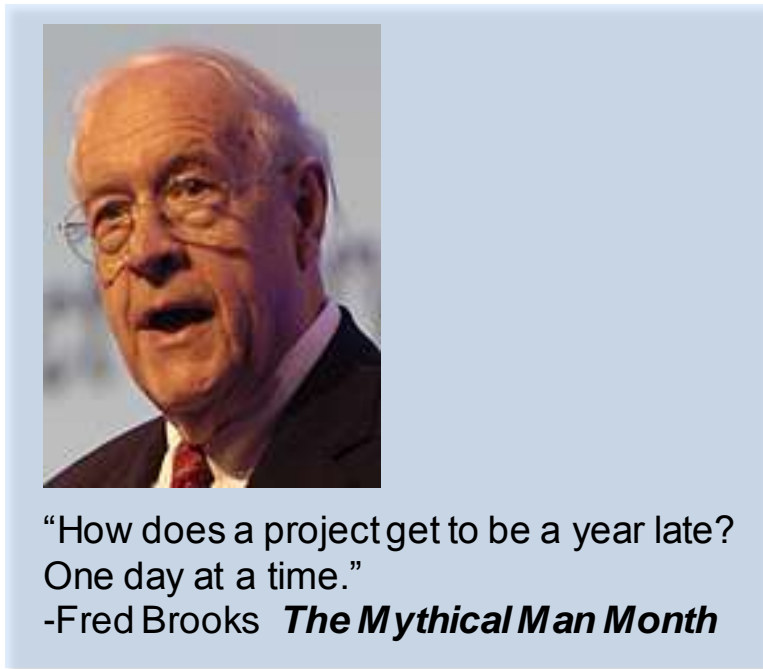
Automation for Agile Program Management

How Agile is your program?

Programs slip a day at a time;

How long does it take to

- recognize an issue ?
- respond to user needs?
- fix a critical vulnerability?



Builds happen constantly.

How current is your data? How credible ? Accurate? How Complete?

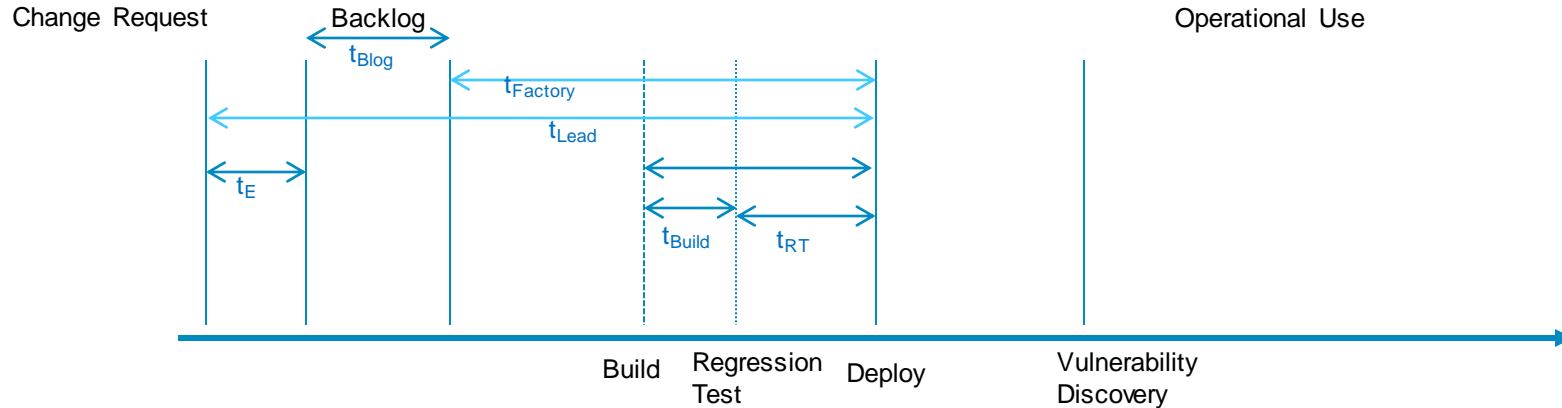
Delivery is driven by automation; **data should also be automated.**

Some Lead Time Definitions

There are many “Lead times” in a DevSecOps development system.

- Averages obscure variation, transfer times, and type of work.
- **Zero Defect** and No Wait times provide a baseline for a minimum practical lead time.

Some useful Lead times are as follows:



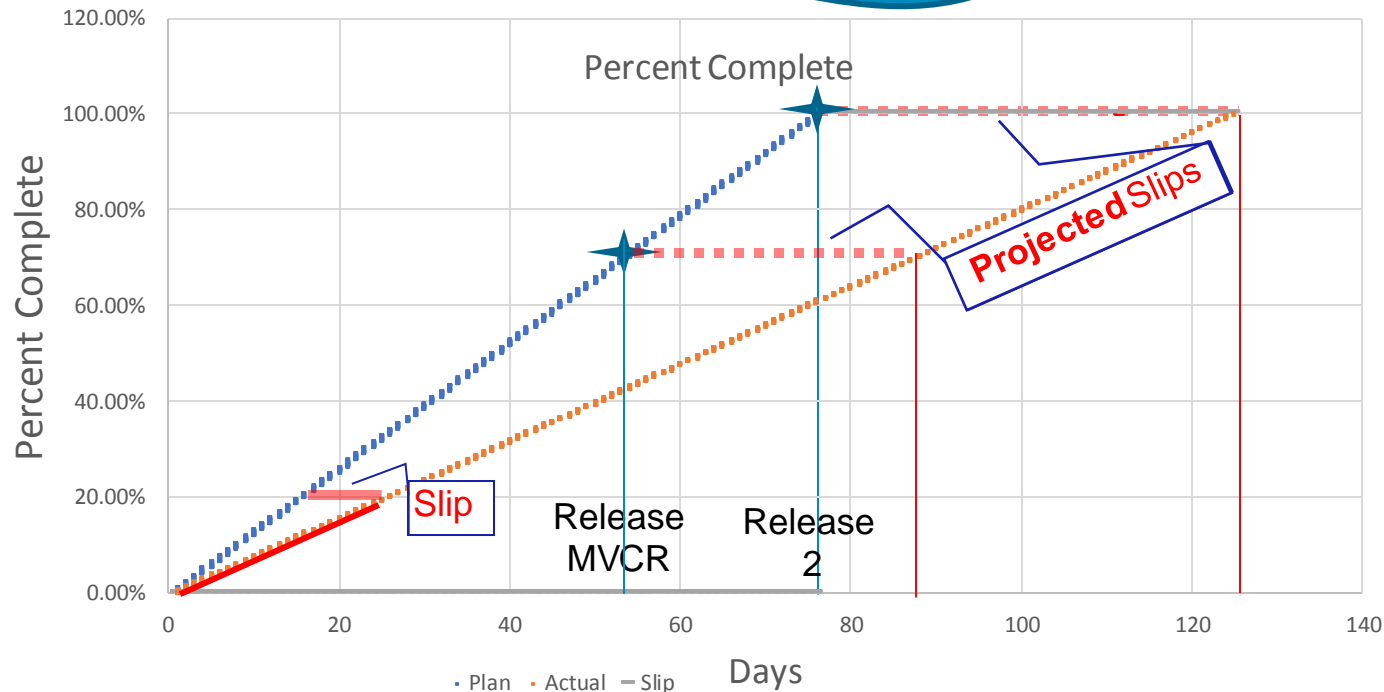
t_{Lead} Lead time
 t_E Evaluation time
 t_{Blog} Backlog time

t_{Build_0} No Rework Build time
 t_{RT_0} No Rework Regression Test time

Behind by 2 weeks, When Will the Next Release Deploy? (To where?)

Straight Line Projections
(Earned Value)

Necessary Data and Information



\bar{t}_{lead} - lead time

$\bar{t}_{Factory}$ - factory (cycle) time

$\frac{Throughput}{Staff\ Hours}$ - output rate

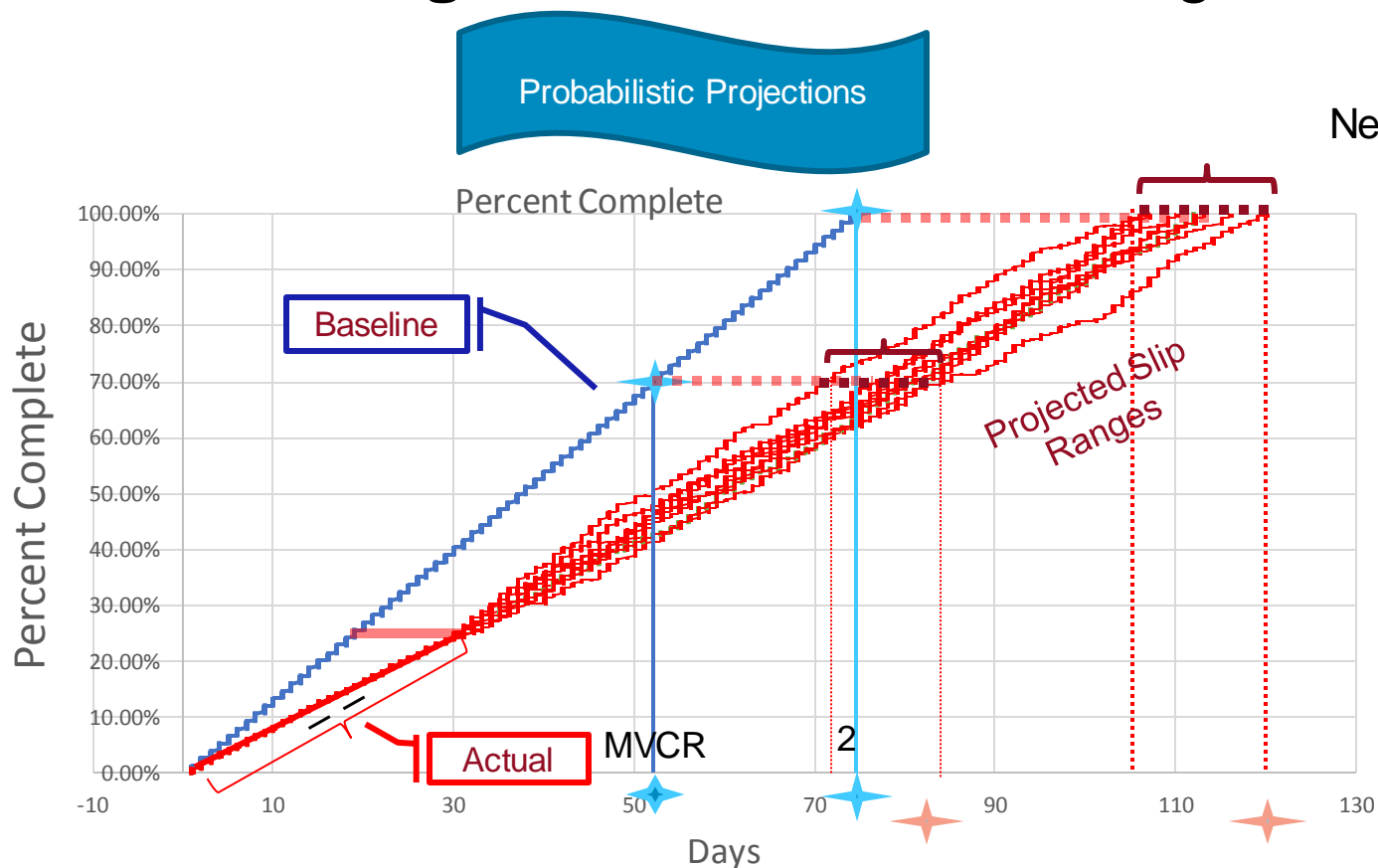
Roadmap

WBS (work packages)

IMS (time sequenced work)

This provides a reasonable estimate, but not a **probability** of success. It is **manual** (pivot tables?), uses last month's data, assumes all work is (more-or-less) the same, and doesn't scale well with multiple pipelines.

Understanding Variation Enables Range Confidence



Necessary Data and Information

Distributions for

$t_{lead}^i, t_{factory}^i, t_{deliver}^i$

Staff Hours(t)

Roadmap

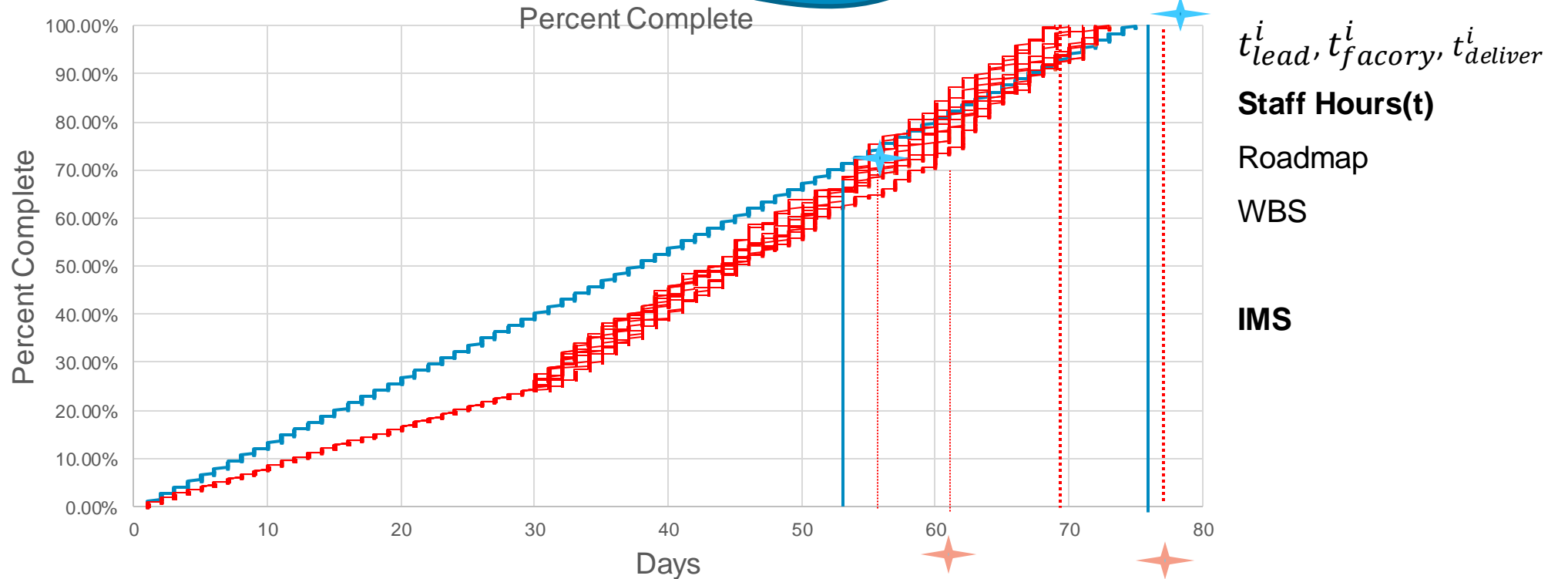
WBS

IMS

Using distributions to capture and analyze variation instead of using averages requires not only maintaining a lot more data but also a lot more work!

Adjust Intervention Parameters to Explore Date Targets

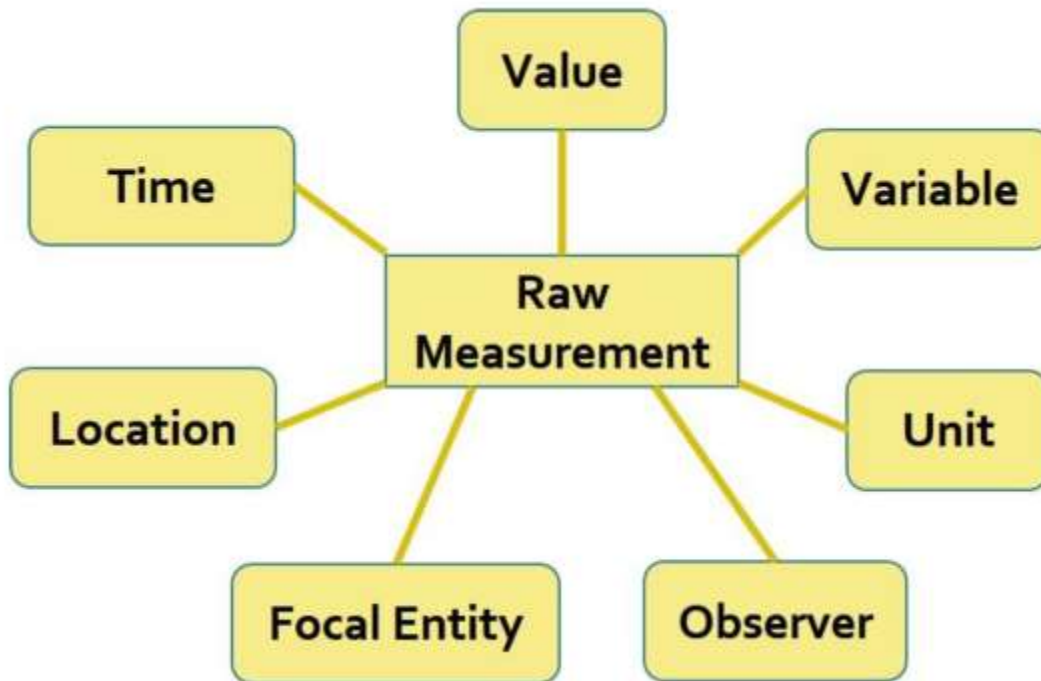
Probabilistic Projections with
“what if” scenarios



Monte Carlo “what if” analysis requires a lot of data and a lot more computation!

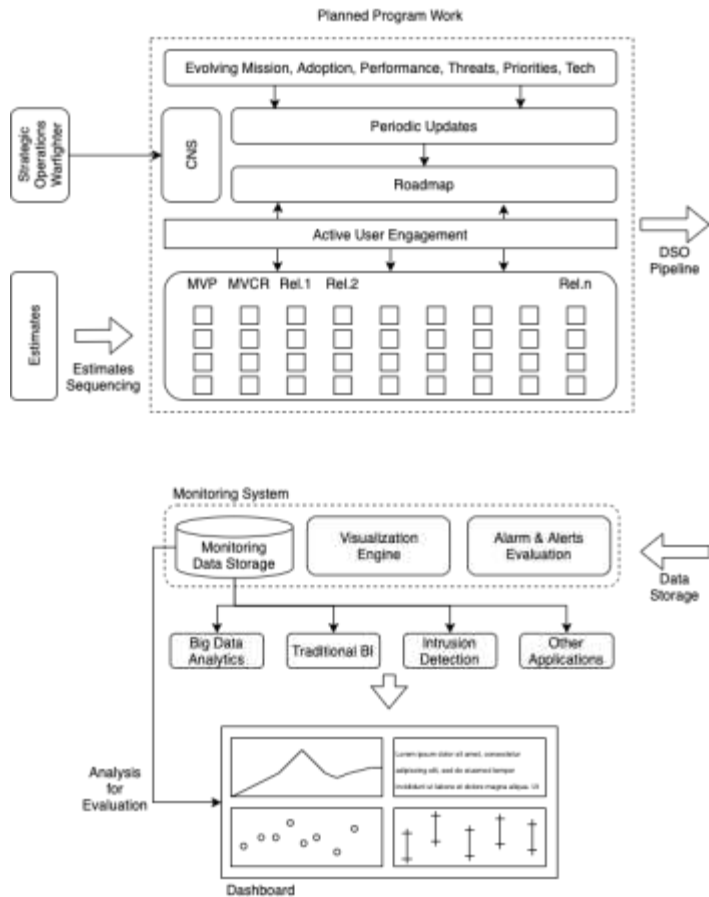
Prototype DSO Measurement

Measurement: *"A set of observations that reduce uncertainty where the result is expressed as a quantity."* - Douglas Hubbard

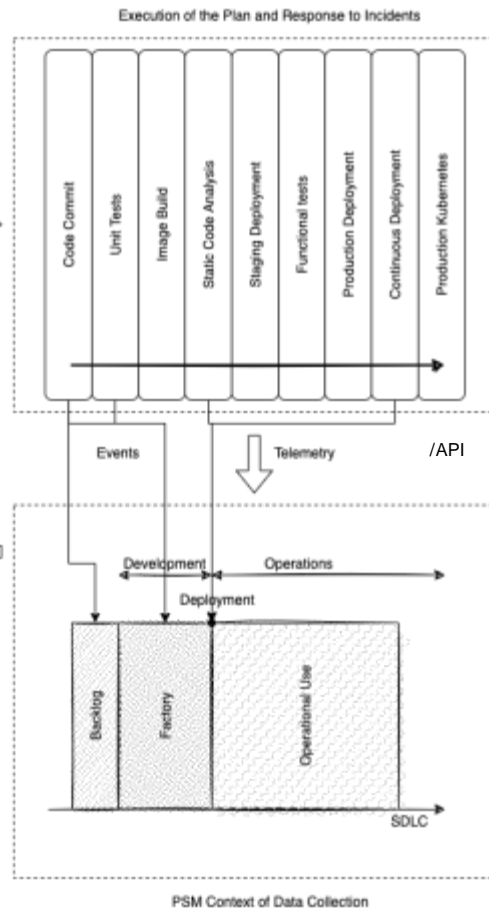


Data Collection Context, the Parts

Managed with Jira, Gitlab, Rally



Factory Pipelines



Planned work includes the WBS, work packages, work sequencing, and estimates.

Work packages **execute** plan development stages. Tools trigger events (time stamps, package labels).

Data is collected and **transformed** for storage.

The **warehouse** loads the data and provides the interface for analysis and dashboards.

From the Roadmap through the Pipeline

Work Completion <https://youtu.be/X-R1mIZ3sPk>

ace-devsecops > Rust Project > Milestones > Feature - Additional Display Function

Closed Milestone May 17, 2021–Jun 7, 2021

Edit Promote Reopen milestone Delete

Feature - Additional Display Function

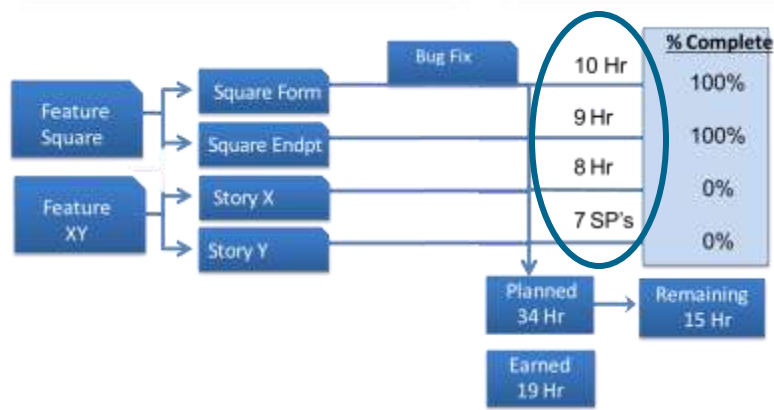
This function should provide the capability for a user of the web service to provide a value, and have the value of its square returned.

Issues 5 Merge Requests 4 Participants 1 Labels 0

Unstarted Issues (open and unassigned) 0

Ongoing Issues (open and assigned) 0

Completed Issues (closed) 5



100% complete >>

Start date May 17, 2021 Edit

Due date Jun 7, 2021 (Past due) Edit

Issues 5 New issue
Open: 0 Closed: 5

Time tracking No estimate or time spent

Merge requests 4
Open: 0 Closed: 0 Merged: 4

Releases
None

Reference: ace-devsecops/rus...

Web Application - Route for Square

#6

Library Test - Square Function

#5

Web Application Feature - Call and Display Square Function

#4

Web Application Feature - User Input

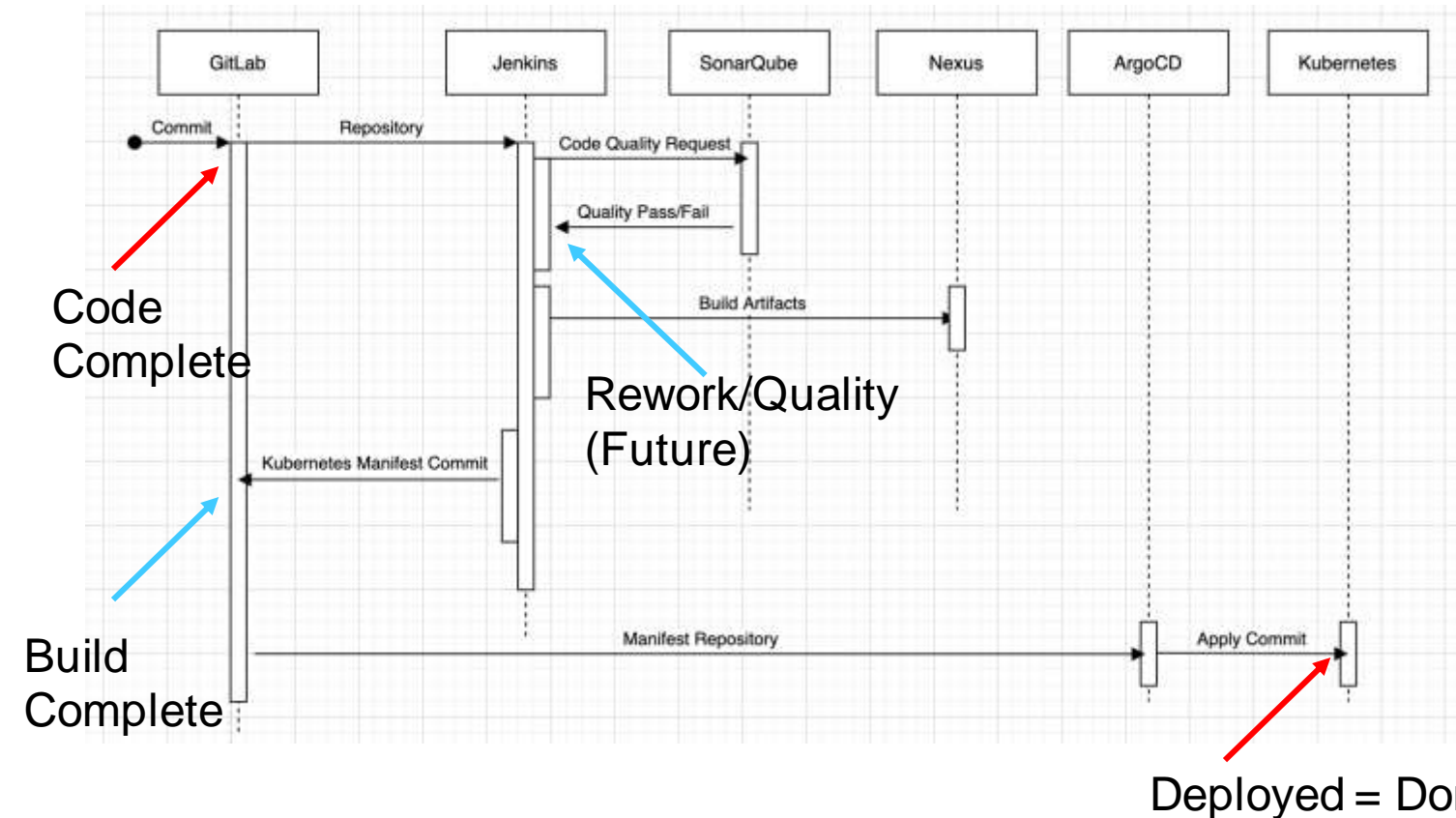
#3

Library Feature - Square Function

#2

Follow Work through the Pipeline

Extracting metrics <https://youtu.be/u96OFTXgr0g>



Date is collected from key events.

The data specification is on the following slide.

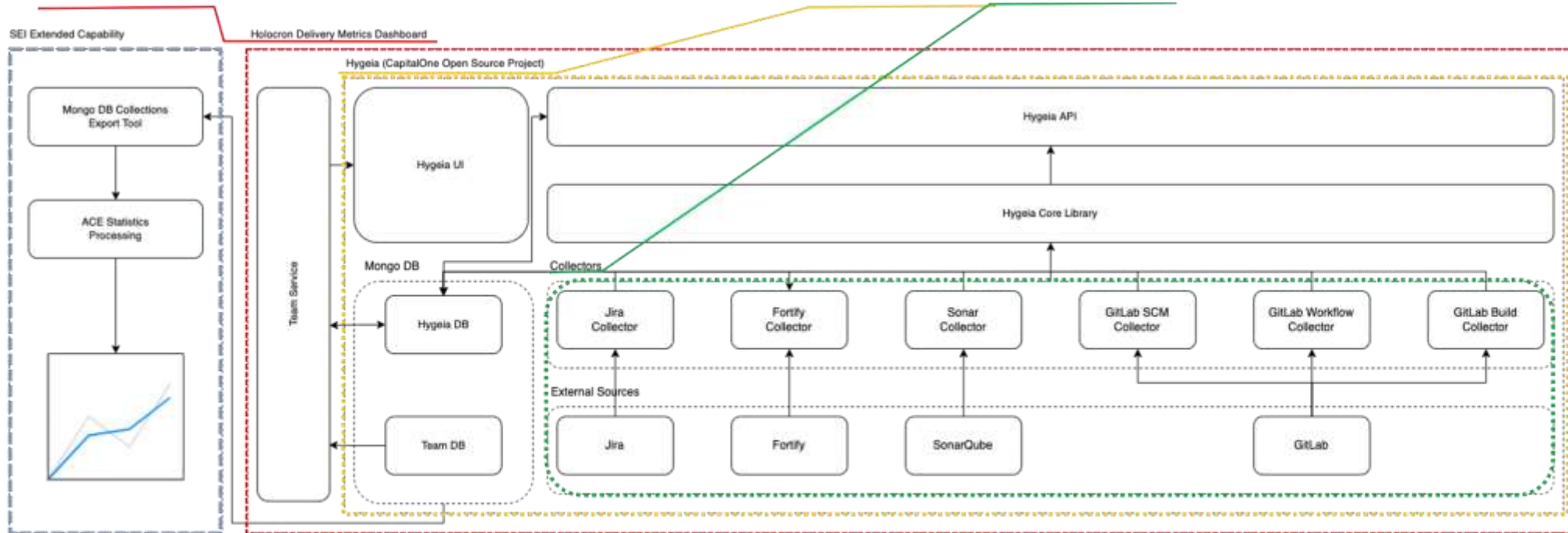
Use Labels to connect WBS, RoadMap, and Backlog to work packages.

- Lead Times
- Estimated Dates
- Actual Times

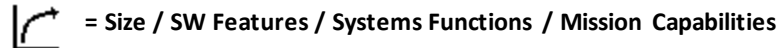
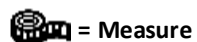
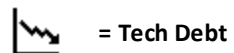
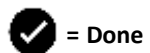
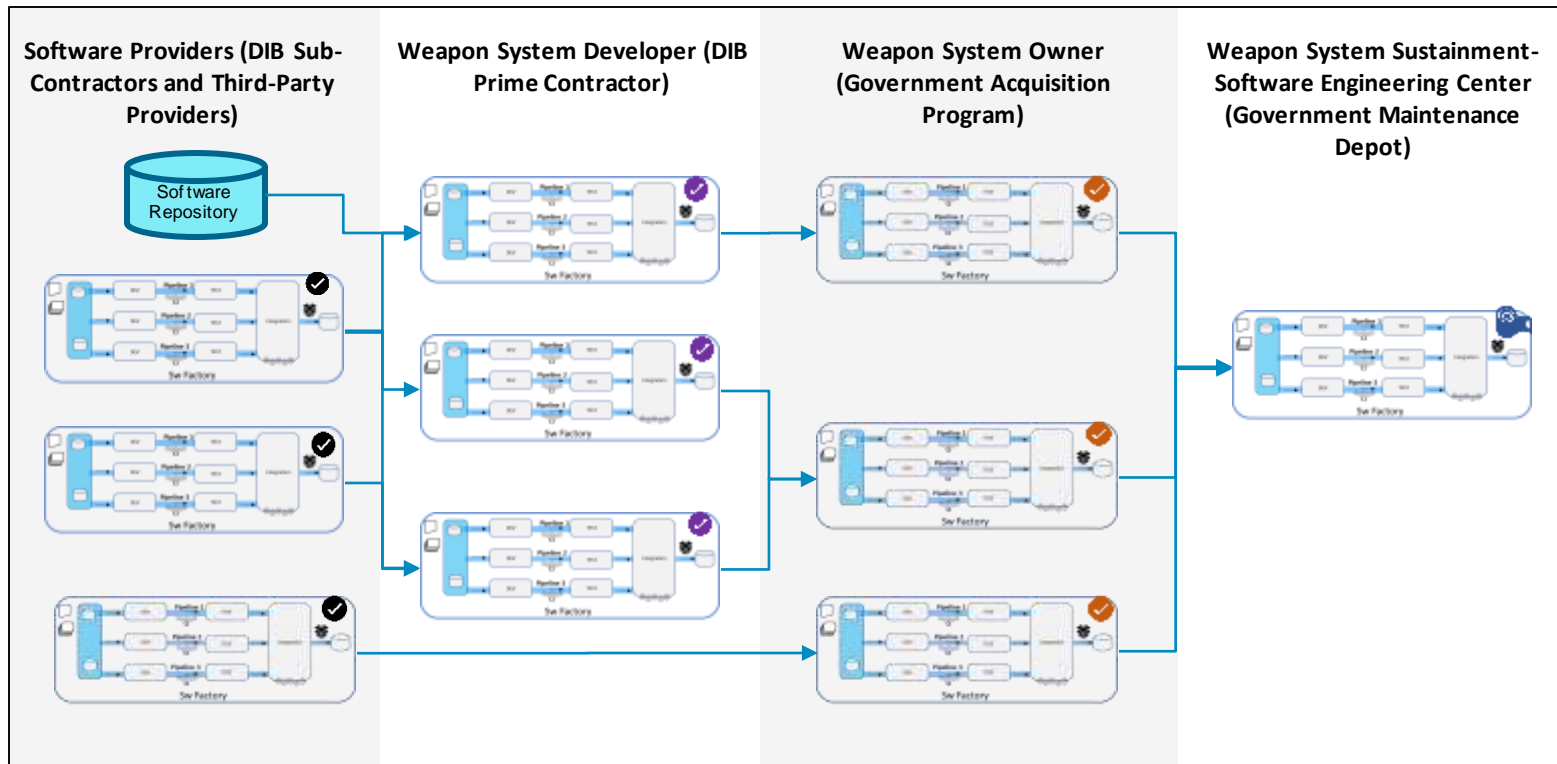
Collect, Store, and Visualize

Apply on local development pipeline (instrumenting a local research project)

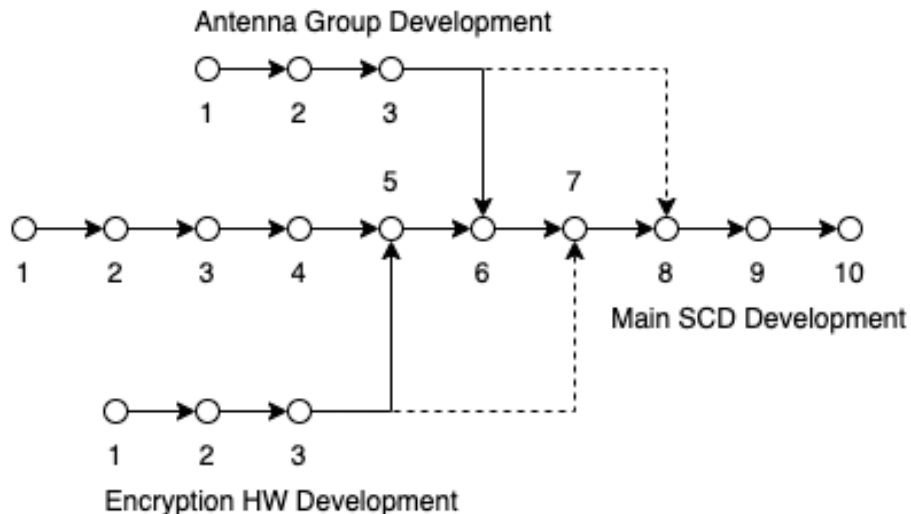
Holocron provided by Platform One, uses **Hygeia Collectors**



Scale Up! Pipeline of Pipelines



Pipeline of Pipelines PoPs Workflow Network Example



Model a fictitious device that captures characteristics of a real project dependencies between hardware and software capabilities.

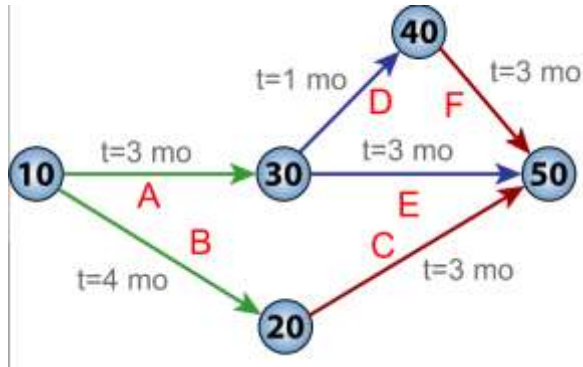
Different pipelines produce dependencies used to model schedule, cost, and technical performance risks resulting from production variation, accumulated variance, and rework.

All nodes are pipeline activities, arrows are lead times. Nodes 5,6,7, and 8 are integration or test points.

Little's Law assumptions are strongly violated except for **some** linear pipeline segments.

Typical Flow Metrics do not accommodate rework, merges, or multiple entry points

Multi-Pipeline Projections: Approach

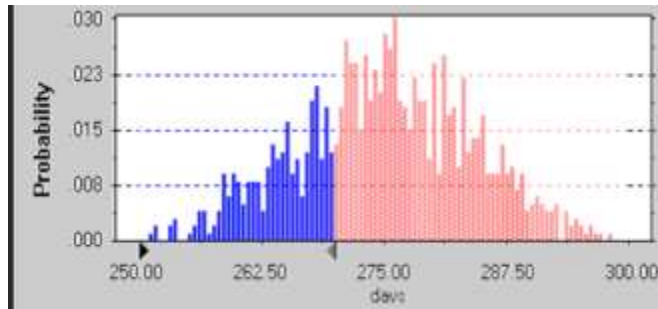


Approach

- Trace work item through development steps
- Identify blockers and integration points
- Probability of completion date

Data from DSO pipeline and other sources

- Product state node structure (capability based WBS, product dependencies, workflow)
- For each Pipeline obtain empirical data for
 - **Effort Rate** and **variation** (by skill?)
 - **Production Rate** and **variation** by **work type**
 - **Primary work** and **Rework** by activity
 - **Defect Rates** and **fix latencies** (build, test)



Observations and Lessons Learned

Measurement tools are too siloed to work together (this is related to a finding from our Digital Engineering work).

Averages don't support statistical modeling or high priority changes. We need distributions.

We need more specific lead time measures for process steps and baselines for zero rework lead times. (total time until test completes is a candidate quality proxy).

Work packages need to be categorized (bug, enhancement, ...) not only because different types of work have different characteristics, but also to gain insight into process health. (a bug vs new capability, vs process sustainment).

Typical flow metrics don't appear to apply to the pipeline-of-pipelines because of branching and other assumptions violations.

Summary

We identified a problem and validated with prototypes and subject matter experts.

We demonstrated

- Practices and tooling needed to instrument the DevSecOps pipeline
- Modeled extensions to the PoPs environment
- Showed limitations in current metrics thinking

Ready to begin transition!

Call to Action

Would you benefit from continuous updates to status and projections?

Are you using DevSecOps tool chains, issue trackers, and workflow management?

Can you share process data and discuss results?

Will you participate in our quarterly research review?

We can help!

- Share out Program Management Measurement White Paper
- Specify information, data, and displays for your program management
- Recommend approaches and tools to get started
- Evaluate your results for effectiveness

Team Photos (Big Team)



William Nichols
Principal Investigator



Luiz Antunes
DevOps Engineer



Rob McCarthy
DevOps Engineer



Chris Miller
Title goes here and can be several lines long.



Julie Cohen
Senior MTS



Melissa Ludwick
MTS



Akia Williams
Senior Administrative Assistant

Thank You!

If you would like to work with us, or if you have any questions, please contact

Bill Nichols info@sei.cmu.edu