Practical Software and Systems Measurement Continuous Iterative Development Measurement Framework

Part 2: Measurement Specifications and Enterprise Measures

Version 2.1
April 15, 2021

Developed and Published by Members of:

Practical Software & Systems Measurement

Product No.
PSM-2021-03-001

National Defense Industrial Association

International Council on Systems Engineering

Product No.
INCOSE-TP-2020-001-06

Editors:

Cheryl L. Jones
US Army
cheryl.l.jones128.civ@mail.mil

Geoff Draper
L3Harris Technologies
geoff.draper@l3harris.com

Bill Golaz
Lockheed Martin
willliam.h.golaz@lmco.com

Paul Janusz
US Army
paul.e.janusz.civ@mail.mil

Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

PSM Product Number: PSM-2020-06-001

INCOSE Product Number: INCOSE-TP-2020-001-06

Copyright Notice:

For this document, each of the collaborative organizations listed on the cover page is the sole manager of their products and services and are the only parties authorized to modify them. Since this is a collaborative product, modifications are managed through the participation of all parties.

General Use: Permission to reproduce, use this document or parts thereof, and to prepare derivative works from this document is granted, with attribution to PSM, NDIA, and INCOSE, and the original author(s), provided this copyright notice is included with all reproductions and derivative works.

Supplemental Materials: Additional materials may be added for tailoring or supplemental purposes if the material developed separately is clearly indicated. A courtesy copy of additional materials shall be forwarded to PSM (psm@psmsc.com, attention: Cheryl Jones). The supplemental materials will remain the property of the author(s) and will not be distributed, but will be coordinated with the other collaboration parties.

Author Use: Authors have full rights to use their contributions with credit to the technical source.

Supplemental Notice from INCOSE: This work is an Affiliate Product per INCOSE Policy TEC-107 INCOSE Technical Product Development & Commercialization (26 October 2018). It is a technical product developed outside the INCOSE product development process and was made by INCOSE members in cooperation with PSM and NDIA; then approved by INCOSE to be distributed from INCOSE central channels. The authors own the copyright and take primary responsibility for proper branding, intellectual property, content quality and appropriate citations with INCOSE oversight based on this policy & related procedure.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

CONTENTS

EXECUTIVE SUMMARY **4**

8. MEASUREMENT SPECIFICATIONS **5**

8.1 AUTOMATED TEST COVERAGE (PRODUCT OR ENTERPRISE MEASURE) 5
8.2 BURNDOWN (TEAM, PRODUCT, OR ENTERPRISE MEASURE) 12
8.3 COMMITTED VS COMPLETED (TEAM, PRODUCT, OR ENTERPRISE MEASURE) 15
8.4 CUMULATIVE FLOW (TEAM, PRODUCT, OR ENTERPRISE MEASURE) 19
8.5 CYCLE TIME/ LEAD TIME (TEAM OR PRODUCT MEASURE) 24
8.6 DEFECT DETECTION (TEAM, PRODUCT, OR ENTERPRISE MEASURE) 27
8.7 DEFECT RESOLUTION (TEAM OR PRODUCT MEASURE) 31
8.8 MEAN TIME TO RESTORE (MTTR) / MEAN TIME TO DETECT (MTTD) (PRODUCT OR ENTERPRISE MEASURE) 35
8.9 RELEASE (OR DEPLOYMENT) FREQUENCY (PRODUCT OR ENTERPRISE MEASURE) 39
8.10 TEAM VELOCITY (TEAM MEASURE) 45
8.11 PRODUCT VALUE (TEAM, PRODUCT, OR ENTERPRISE MEASURE) 48

9. ENTERPRISE MEASURES **61**

9.1 CHALLENGES FOR CID ENTERPRISE MEASURES 62
9.2 EXAMPLE ENTERPRISE MEASUREMENT INDICATORS 65

LIST OF FIGURES

Figure 1: Automated Test Coverage (Project Level) 6
Figure 2: Automated Test Pass/Fail Status 7
Figure 3: Code Coverage from Automated Testing 8
Figure 4: Automated Test Coverage (Enterprise Level) 9
Figure 5: Release Burndown 13
Figure 6: Stories Completed versus Committed 16
Figure 7: Program Completed versus Committed 17
Figure 8: Cumulative Flow Diagram 20
Figure 9: Notional CFD Diagram 21
Figure 10: Workflow by Period and Rolling Average 22
Figure 11: Cycle Time: Closed Issues 25
Figure 12: Defect Terminology 28
Figure 13: Defects Detected versus Resolved 31
Figure 14: Cumulative Defects Detected vs. Cumulative Defects Resolved 32
Figure 15: Defect Resolution Lag Time 32
Figure 16: Operations Outage Summary 36
Figure 17: Iterative Development 39
Figure 18: Product Iterative Releases (Conceptual) 40
Figure 19: Release Duration for Product Tango 42
Figure 20: Product Release Frequency 42
Figure 21: Team Velocity 46
Figure 22: Stakeholder Attribute Value 52

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Figure 23: Alignment of Enterprise Measures 61
Figure 24: Burndown - In Progress Release Report 66
Figure 25: Aggregate Indicator - Stories Committed vs. Completed 67
Figure 26: Stoplight Indicator - Estimate Accuracy 68
Figure 27: Software Estimate Accuracy Summary 68
Figure 28: Workflow Indicator 69
Figure 29: Cumulative Flow Stoplight 69
Figure 30: Mean Time to Repair 70
Figure 31: MTTR Stoplight Indicator 71
Figure 32: Aggregate Software Productivity - Acceleration 72

LIST OF TABLES

Table 1: Defect Detection by Release 28
Table 2: Defect Resolution Lag Time 29
Table 3: Defect Resolution Lag Time 33
Table 4: MTTR Statistics 36
Table 5: Product Release Averages 41
Table 6: Release Frequency and Labor Hours 41
Table 7: Sample Acceleration 46
Table 8: Sample Stakeholder Attribute Values 51
Table 9: Challenges for Collecting and Reporting CID Measures at the Enterprise Level 63
Table 10: Measurement Specification Mapping and Examples 65
Table 11: Acceleration 72

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

EXECUTIVE SUMMARY

This report provides recommendations for the measurement of continuous iterative developments (CID). The report includes a Practical Software and Systems Measurement (PSM) CID measurement framework detailing common information needs and measures that are effective for evaluating CID approaches. The information needs address the team, product, and enterprise perspectives to provide insight and drive decision-making. The framework also identifies and specifies an initial set of measures that have been identified as being practical measures to address these information needs.

This guidance is intended to be used by team, program, and enterprise personnel who are implementing CID approaches, as a reference for common, practical measures that can be utilized. The measures a program or enterprise chooses to implement and collect will be tailored based on alignment with its information needs and objectives, so they may differ from those described here. The measures presented are intended to be tailored and adapted to the development approach and environment.

Version 1.05 detailed potential information needs and measures that are common to CID approaches, and an initial set of ten measurement specifications that were prioritized by user surveys as highest value. This Version 2.0 (draft) is a review release, to allow review of added material that has been researched and developed by the CID working group. The new materials include information on measuring:

- Product value (Part 2, section 8.11)
- Enterprise measurement (Part 2, section 9)
- Software assurance (Part 3, section 10)
- Technical debt (Part 3, section 11)

Part 1 of this report includes a series of diagrams and an ontology to describe the development approaches and terminology used. It also includes an "Information Category-Measurable Concept-Measures" (ICM) Table detailing potential information needs and measures for CID developments. Additional potential measures will be added in future releases, as described in Section 6, Next Steps.

For the highest priority measures, sample measurement specifications have been developed that detail the identified measures. These are included in this paper, Part 2, along with a discussion of how to use these measures for enterprise decision making. Part 3 of the paper separately extends the main CID paper with detailed information and guidance on Software Assurance and Technical Debt.

We invite your comments on this material, and your participation in future updates addressing additional measures and guidance.

This report is intended to be methodology and approach-agnostic and is written so that it may be adapted to organizational needs. Different methodologies and tools may use different terminology than defined in this report.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8. Measurement Specifications

8.1 Automated Test Coverage (Product or Enterprise Measure)

Measure Introduction	
Description	In an iterative development approach, it is important not only to efficiently verify new features but to ensure prior functionality is not impacted. Doing so manually can be time-consuming. Typically, code coverage is verified primarily in structural (white box) testing at the unit level, and requirements are verified primarily in functional/system test. Efficiency and throughput can be enabled by automated test suites executed at multiple levels (unit level, functional level, regression testing). The extent to which automated testing is implemented is a business decision depending on objectives and constraints, such as velocity, quality, and cost vs. benefit. It may not be feasible or desirable to automate all testing. Projects may set planned test automation objectives, such as 70%-80% coverage based on their cost benefit analysis. Often these automated test suites are integrated directly in the code pipeline and invoked upon each code commit and build, or in nightly regression test batch jobs. (Refer to Figure 2 for context.) Test results (tests passed, tests failed) can be distributed automatically in email so anomalies impacting the code quality and pipeline can be quickly identified and resolved.
Relevant Terminology	Functional Testing: Testing against the requirements or function of the software, without considering the internal implementation. Sometimes termed black box testing. Structural Testing: Testing the internal structure, design, implementation, or logic of software, such as paths, conditionals, or branches through the code. Sometime termed white box testing.

Information Need and Measure Description	
Information Need	How much of the testing is automated? How many tests have been validated and approved? How much credit is given in formal test (e.g., DT/OT) for automated test?
Base Measure 1	Total Requirements *[integer > 0]*
Base Measure 2	Requirements Tested *[integer ≥ 0]*
Base Measure 3	Requirements Tested Through Automation *[integer ≥ 0]*
Base Measure 4	Requirements Tested Manually *[integer ≥ 0]*
Base Measure 5	Code Constructs (e.g., classes, conditionals, files, lines, packages) *[integer > 0]*
Base Measure 6	Code Constructs Tested by Automated Test *[integer ≥ 0]*
Base Measure 7	Automated Test Cases Passed *[integer ≥ 0]*
Base Measure 8	Automate Test Cases Failed *[integer ≥ 0]*
Derived Measure 1	Requirements Not Tested = (Total Requirements) – (Requirements Tested Through Automation) – (Requirements Tested Manually) *[integer ≥ 0]*
Derived Measure 2	Percentage Requirements Tested Through Automation = (Requirements Tested Through Automation) / (Total Requirements) * 100 *[percentage]*
Derived Measure 3	Percentage Requirements Tested Manually = (Requirements Tested Manually) / (total requirements) * 100 *[percentage]*
Derived Measure 4	Percentage Requirements Not Tested = (Requirements Tested Not Tested) / (total requirements) * 100 *[percentage]*
Derived Measure 5	Percentage Code Constructs Tested = (Code Constructs Tested by Automated Test) / (Code Constructs) * 100 [percentage] *(for each code construct) [percentage]*

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Indicator Specification	
Indicator Description and Sample	Figure 1 depicts the percentage of project requirements that are verified by automated vs. manual testing over time. In this example, the project set a planned objective for 70% automation, and ultimately met and exceeded that objective. Percentages are used rather than absolute values to facilitate comparisons across projects. The total number of requirements changes over time as more requirements are developed, and are plotted on the secondary axis to enable consideration of the scale and complexity of the test automation effort. Tradeoff decisions can be made on the benefit of investing further program effort to develop new automated test cases to increase coverage. This may include estimating the net impact on program throughput, quality, or cost.

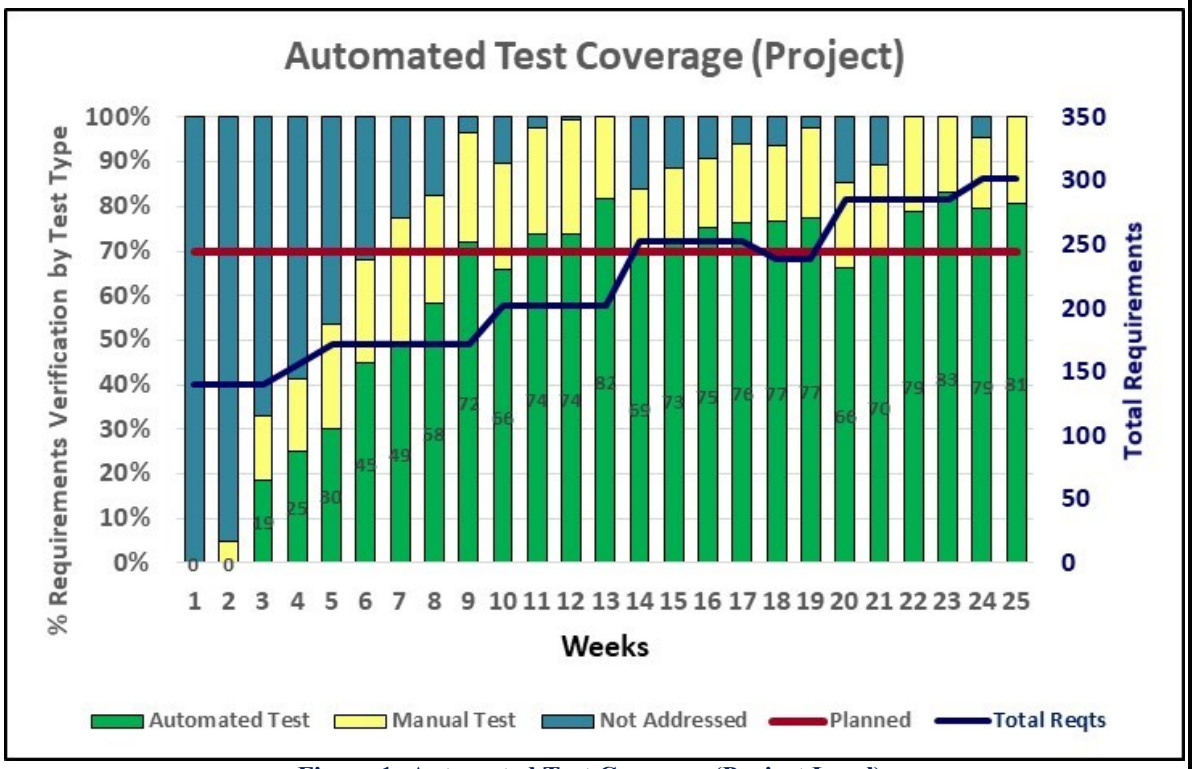


Figure 1: Automated Test Coverage (Project Level)

At project startup an initial requirement set is established that evolves iteratively (with additions, modifications, deletions) across the project life based on collaboration with the product owner and other stakeholders. Test cases (automated and manual) are developed to verify requirements as they are implemented. By iteration 9, the automated test suite is verifying over 70% of requirements, supplemented by manual test cases that verify nearly all project requirements. In iteration 18, the product owner deleted a capability from the backlog and requirements count was reduced. Over time, additional automated tests are developed that increase automated coverage while reducing the dependence on manual testing, although both are supplemented regularly as new requirements are added. The project has sustained its automated test suite to generally meet the project objective of 70%-80% automated test coverage.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Effectiveness of automated testing should be monitored. The pass/fail success status of automated tests is often available from automated test tools, as illustrated below in Figure 2, so anomalies breaking the code pipeline can be quickly detected and resolved. The quantity of requirements covered in automating testing is depicted in the amplitude. Requirements that failed an automated functional test are shown in red, indicating quality of the pipeline over time. Some tools may also provide additional information, such as requirements that were skipped, or the requirements with no automated test.

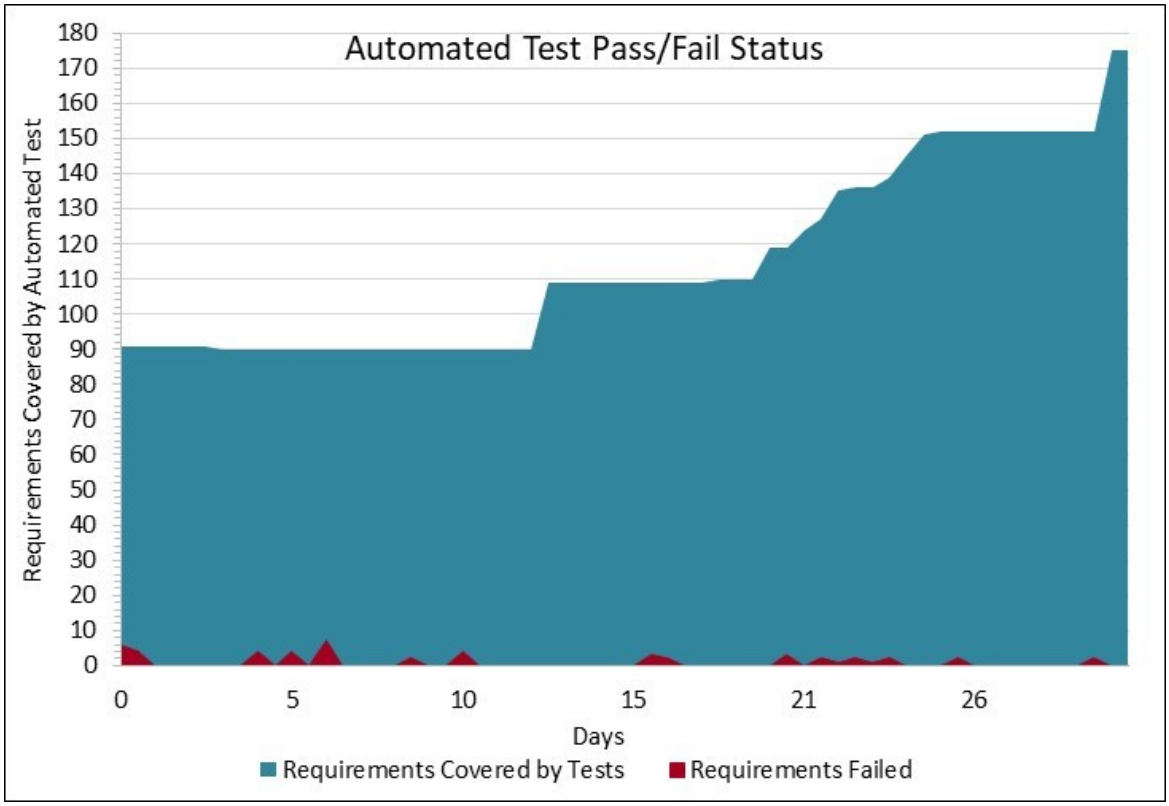


Figure 2: Automated Test Pass/Fail Status

This automated report from the program test tool indicates a low number of requirements (<5) over time that failed automated testing. All test failures are investigated. Some of the test failures are due to enhancing the automated test scripts to verify new requirements as they are added, others are the result of regression test failures where baseline product functionality was impacted by new enhancements, but this quickly stabilizes as the product development baseline matures.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

The extent of code structural coverage from automated (white box) testing can increase confidence in development baseline quality. In Figure 3 test coverage is collected for each increment and depicted by trends for % coverage of structural code constructs (classes, conditionals, files, lines, packages). The extent of coverage can indicate the risk or confidence in code quality, suggest a need for additional testing, or the potential risk of incurring defect escapes.

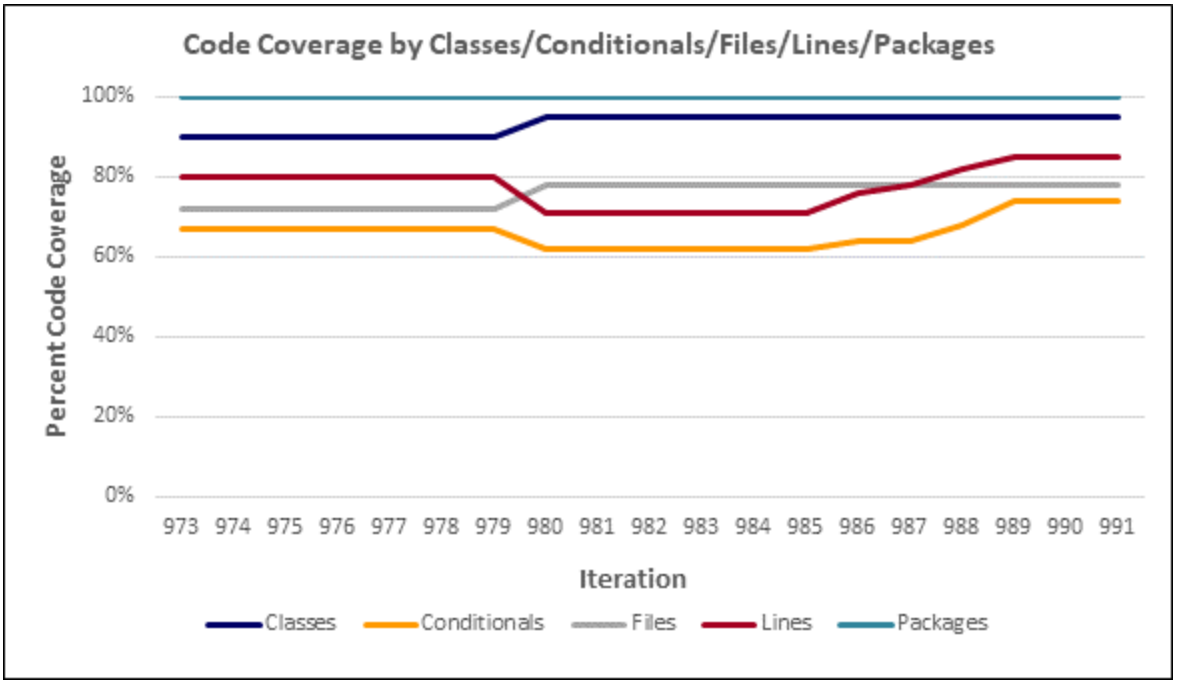


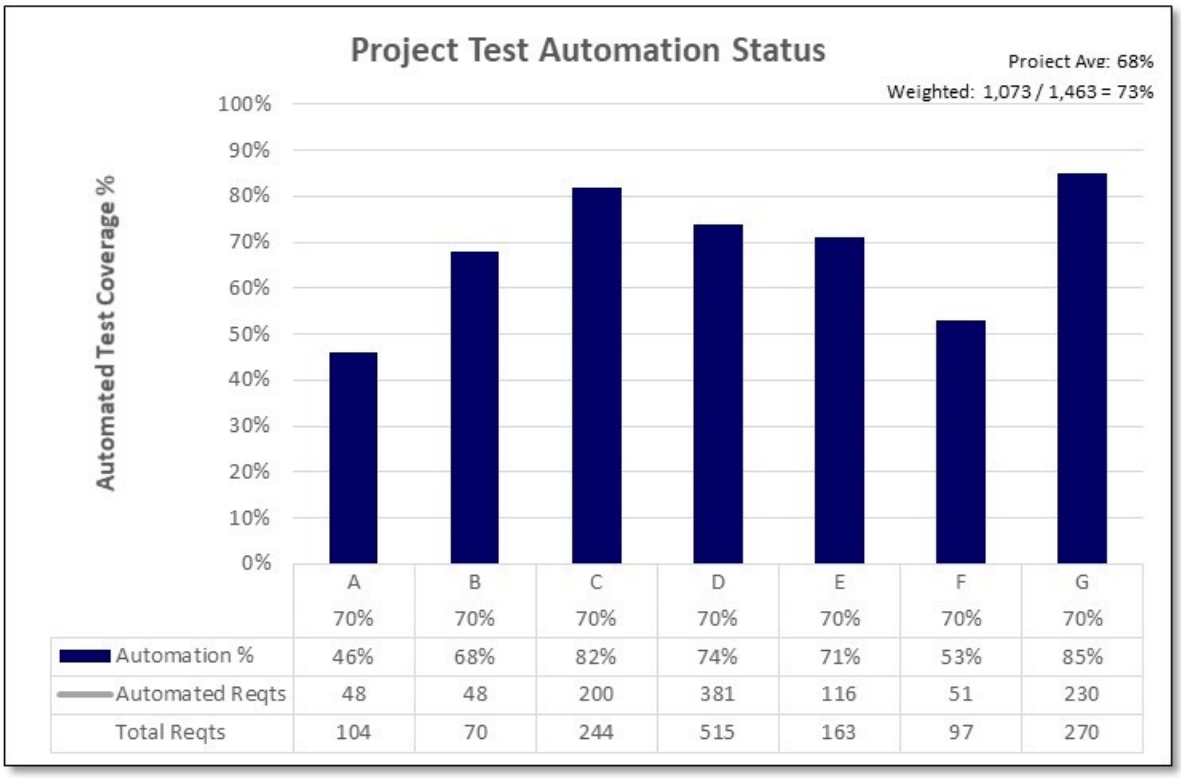
Figure 3: Code Coverage from Automated Testing

100% of packages and 95% of classes are addressed by automated tests. 85% of the code (lines of code) and 75% of branches are currently exercised; coverage dropped in iteration 980 (to 70% of code, 65% of branches) as new functionality was added, but has continued to grow in subsequent releases as the automated test suite was expanded to address these enhancements. The project has set a target for ≥ 80% of code and branches exercised in automated testing, so the test suite is being enhanced for additional logic test cases focusing on the most risky or complex modules.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Indicator Description and Sample (continued)

At the enterprise level, the extent of automated testing utilized across projects can be monitored, as reflected in Figure 4. The enterprise may set business objectives for the extent of automated testing across projects (e.g., 70%), subject to project-specific characteristics and constraints.



	A	B	C	D	E	F	G
	70%	70%	70%	70%	70%	70%	70%
Automation %	46%	68%	82%	74%	71%	53%	85%
Automated Reqts	48	48	200	381	116	51	230
Total Reqts	104	70	244	515	163	97	270

Figure 4: Automated Test Coverage (Enterprise Level)

Automated test coverage percentages are collected from projects and aggregated at the enterprise level to monitor the success of implementing automated testing. Measures are displayed for each project in both relative (%) and absolute terms (Requirements Verified). Absolute values are used for context in evaluating the overall impact of the project automated test coverage; larger projects may have greater challenges in scope but also more resources available to realize the benefits of automation. Some projects are early in their development cycle and development of automated test cases are still in work. Overall, the project average is 68% automation, but when weighted by the number of requirements verified the coverage is 73% due to the higher impact from larger projects. Analysis and actions at the organizational level will depend on the characteristics of the individual projects, the extent to which performance and quality measures are impacting objectives, and the extent to which they may be positively impacted by investing in additional automation.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Analysis Model	Automated Test Coverage (Project Level): • What percentage of functional requirements are verified with automated testing? Is each requirement fully covered by the automated testing, or are some aspects not verified? • Any requirements not verified automatically must be verified manually, which can impact productivity, schedule, and resources. Apply decision tradeoffs for the cost vs. performance benefit of investing effort to expand the extent of automated test coverage. Automated Test Pass/Fail Status: • Are automated tests completing successfully, or are there anomalies impacting the code pipeline that should be investigated? • Automated tests are typically conducted regularly as part of the code and unit tests in the code development pipeline, such as upon each code commit or in nightly regression tests. Summary test reports can be automatically generated and distributed by the automated test tools. 100% success of automated tests passing is often a criterion for advancing the code baseline to production. Discrepancies could be in the code, or in the test cases themselves, but either should be investigated. Code Coverage from Automated Testing: • How much of the code structure is covered by the automated test suite? Which parts of the code are not covered (e.g., any safety critical code, interfaces, interoperability requirements)? • Code coverage is a tradeoff between investment, risk, and return; although 100% coverage may be desirable, that might not be practical within available environments, resources, interfaces, and constraints. Automated Test Coverage (Enterprise Level): • What is the extent of automated testing conducted across the organization's projects? What benefits to organizational performance (e.g., cycle time, quality, throughput) are enabled by effective automated testing? Automated testing is a primary enabler for achieving efficiency, quality, and cost savings at both the project and organizational levels. Organizations should monitor automated test measures in relation to achievement of their desired performance objectives.
Decision Criteria	Automated test coverage alone is not an objective; it is the associated gains in accelerating performance and improving product quality at the project and organizational levels that make investments in automation worthwhile. Automation measures should be evaluated in the context of other performance measures, such as those defined elsewhere in the PSM CID measurement framework. Industry experience suggests that automation in the range of 70%-80% is often beneficial in producing improved performance outcomes, but this may vary by domain or application. If automation measures are lower than planned, or if there are process effectiveness or product quality issues that are impacting objectives, consider root cause analysis and decision tradeoffs to assess the impact and determine if they can be improved by further investments in test automation.

Additional Information	
Additional Analysis Guidance	Test automation and coverage are key elements of achieving faster and more comprehensive releases with higher code quality. These should be used in conjunction with quality measures to ensure the adequacy of testing and achieve acceptable, inherent quality levels. A reasonable goal is to achieve near instantaneous automated test results with acceptable quality. Testing efficiency and speed are closely related to achieving other performance measurement objectives such as lead time, cycle time, and release frequency. Robustness of the testing conducted should also be considered (e.g., stress testing, boundary conditions on valid data inputs). Additional project performance measures, such as effort, schedule, and cost, can be correlated with automated test coverage measures to evaluate the performance benefits (e.g., cost savings, productivity, quality) achieved through automated testing. Alternative thresholds or weighting could be applied to automated test coverage scores based on characteristics of a project or component, such as size, complexity, reuse, criticality, or other dependencies.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Implementation Considerations	Measures for code coverage and requirements coverage are directly available from many automated development tools commonly integrated across the tool chain. However, the emphasis should be on thorough testing sufficient to ensure product quality rather than achieving high code coverage numbers. Code coverage is an important factor, but by itself, is not sufficient to ensure product quality. Automated test cases could focus on areas of high risk, complexity, or dependencies where repeatability or regression testing are important factors, especially in the near term. Relying solely on automated test tools and scripts may not be wholly sufficient to exercise all functionality needed (e.g., user interfaces, databases). It may be necessary to supplement automated test scripts with manual effort to execute additional test cases and validate that the automated test is sufficiently representative of the overall functionality. Automated testing may be conducted at various or multiple points in the workflow, for instance before or after the baseline merge. A best practice is to execute automated test suites nightly or as part of the pipeline following each code commit. For existing systems, the enterprise will need to make a business decision as to whether it is worth the investment to develop automated tests. This will be dependent on the necessary infrastructure to support automated test, the expected lifecycle of the system, the level of updates/regression test typically required, etc. Automated test scripts are a valuable work asset that should be sustained in a manner similar to source code. Test scripts may need to be enhanced or refactored as the product evolves.

Additional Specification Information	
Information Category	Process Performance (Process Effectiveness)
Measurable Concept	Process Effectiveness
Relevant Entities	System, Test cases
Attributes	Amount tested, amount automated tested
Data Collection Procedure	Data is typically collected by automated tools upon execution of test scripts as part of standard pipeline workflows. Results are recorded in team tracking tools. Summaries of test results and coverage can often be provided automatically nightly or upon completion.
Data Analysis Procedure	Data is reviewed and analyzed to ensure adequate quality for each candidate product. Discrepancies in process effectiveness, product quality, or test coverage not meeting threshold targets may indicate updates to code or test scripts are necessary.

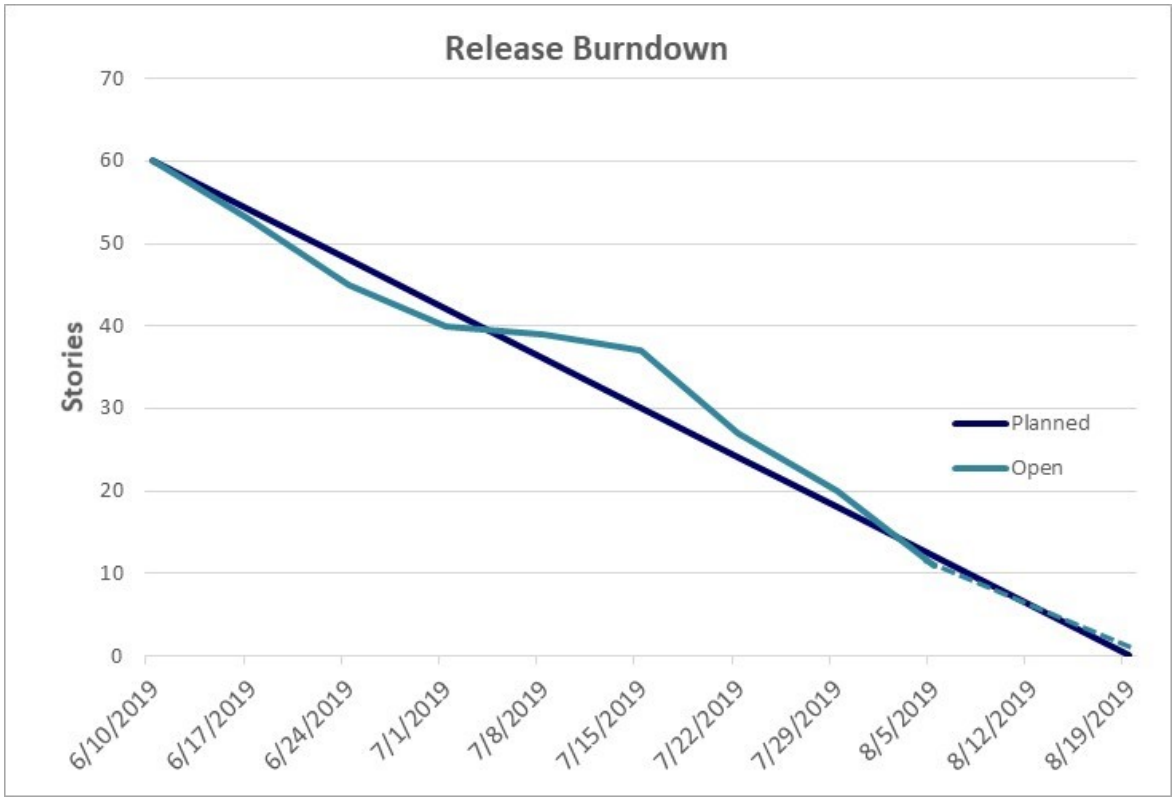
Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.2 Burndown (Team, Product, or Enterprise Measure)

Measure Introduction	
Description	Burndown is used to monitor completed work items (e.g., stories, features, capabilities) vs. planned work items for an iteration, release, or capability. Work items may include design, code, test and all supporting activities (e.g., requirements development, configuration management and quality engineering). Progress toward completing planned work is depicted graphically to provide an indicator of the likelihood of meeting planned goals.
Relevant Terminology	See Section 3 in Part 1: Ontology and Definitions.

Information Need and Measure Description	
Information Need	What is the status of the iteration, release, or capability? Will all the remaining committed work be completed as planned? What are the features/capabilities at risk of not being completed as scheduled? What are the trends in execution relative to plan?
Base Measure 1	Planned Work (integer scale) (e.g., Story Points/Features/Capabilities)
Base Measure 2	Completed Work (integer scale) (e.g., Story Points/Features/Capabilities)
Derived Measure 1	Open Work = Planned Work - Completed Work (e.g., Story Points/Features/Capabilities)

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Indicator Specification	
Indicator Description and Sample	In Figure 5, the teal line represents the number of open stories over time, while the dark blue line indicates the planned burndown. This chart shows a 2-month release, with weekly increments, where stories are completed.  **Figure 5: Release Burndown** At release planning, work items representing 60 stories were committed. While little progress was made during the first week to a planned training event, the teams recovered and is projected to complete the planned work by the end of the release.
Analysis Model	At the team level, the focus is generally on stories or story points open through the iteration. Is the team completing the committed work items? Are they significantly behind or ahead of the burndown plan? Are items blocked? What is the likelihood of meeting the commitment on time? Can additional backlog stories be brought into the iteration? Are teams improving execution over time? At the product level, the focus turns to features or capabilities across releases. At the enterprise level, the focus is generally on capabilities for external releases.
Decision Criteria	At the team level, lack of progress (e.g., not reducing open story points at all over several days) and variances from the plan (e.g., 5%) should be reviewed for action by the team. Data is generally not shared externally to the team. At the product level, variances of over 10% are reviewed for causes of roadblocks and consideration of replanning.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Additional Information	
Additional Analysis Guidance	Use this metric with the velocity metric and other work unit progress metrics (e.g., test progress, cumulative flow). The velocity metric supports the planned story points for each iteration. The actual completed story points from the iteration is an input to the velocity metric. Review with other work unit progress metrics may support an assessment of overall risk and may impact prioritization of work for future iterations. Consider bounds of estimated burndown based on historical performance, e.g., best case, worst case, Monte Carlo analysis.
Implementation Considerations	Estimates are typically based on measures of relative effort, such as user stories, story points, or other validated alternatives based on clear, repeatable operational definitions.

Additional Specification Information	
Information Category	Schedule and Progress
Measurable Concept	Work Unit Progress
Relevant Entities	Product
Attributes	Story Points, Features, Capabilities
Data Collection Procedure	At the team level, story points committed for each iteration are determined at the iteration planning meeting. This value is determined from the velocity metric. Based on the average velocity and other factors (e.g., vacations), the team commits to a number of story points for the next iteration. Work items (e.g., stories, tasks) are selected to match this commitment. Work items are closed when completed and meet their evaluation criteria, and burndown progress is updated daily. At the product level, the features and capabilities committed for each release are determined during release planning. Commitments may be replanned as work is completed and priorities change.
Data Analysis Procedure	For the team, Burndown is analyzed daily for progress/risk and at the end of each iteration to determine if the story points were delivered as committed. The final story points completed value is an input to the velocity metric. For the project, Burndown is analyzed periodically (e.g., monthly, quarterly, by release). For the enterprise, Burndown of capabilities for major events is analyzed.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.3 Committed vs Completed (Team, Product, or Enterprise Measure)

Measure Introduction		
Description	Committed vs Completed is a measure of progress toward completing planned, or expected, features and capabilities. At the team level it may be used to measure progress of each iteration. At the program or organizational level, it can be used to measure overall progress toward a release and completing product development. It may also be used to measure quality of the product by indicating product readiness with respect to expected capability, or functionality.	
Relevant Terminology	Stories Committed	Stories the team has committed to complete within an iteration.
	Features, or Capabilities, or Committed	Features and capabilities committed to the customer by the program to be included in the product.
	Completed Stories, Features, or Capabilities	Stories that have completed their level of verification and validation and have been proven to work as expected.

Information Need and Measure Description	
Information Need	Are Stories, Features, or Capabilities delivered as committed? What are the Stories/Features/Capabilities at risk of not being completed as scheduled?
Base Measure 1	Work Items Committed Each Iteration (integer) (e.g., stories, story points)
Base Measure 2	Work Items Completed Each Iteration (integer) (e.g., stories, story points)
Base Measure 5	Work Items Committed Each Release (integer) (e.g., features, capabilities)
Base Measure 6	Work Items Completed Each Release (integer) (e.g., features, capabilities)
Derived Measure 1	Percent Work Items Completed = (Sum of All Work Items Completed) * 100 / (Sum of All Work Items Committed) for a desired iteration, release, or program (e.g., stories, story points, features, capabilities)

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Indicator Specification	
Indicator Description and Sample	In Figure 6, Stories Committed is graphed as a column for each iteration [dark blue bar] and stories Completed as a column for each iteration [green bar]. Cumulative Percent Stories Completed are also graphed as a line chart across iterations (secondary axis). The indicator may be aggregated for a release, set of features, capability, or a complete project to provide progress toward product completion.

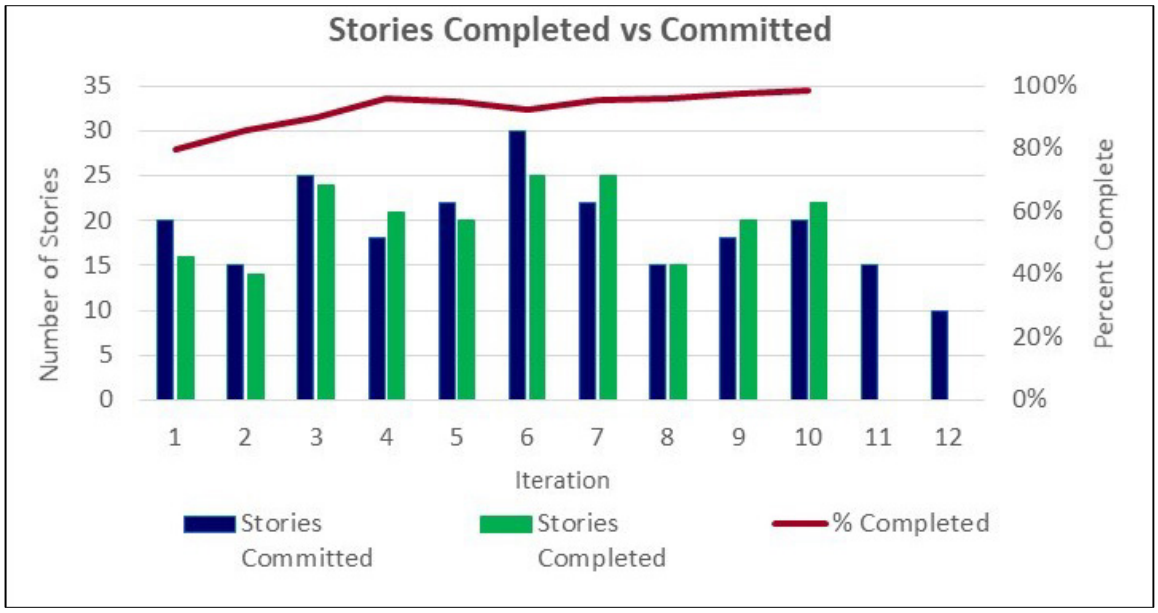


Figure 6: Stories Completed versus Committed

Iterations 1, 2, 3, 5 and 6 did not complete expected stories. During iterations 1 and 2, the team was forming and learning to work together. Iteration 3 completed close to expected stories. Iterations 4, 7, and 9 completed above expected stories. The team was working together and attempting to catch up on the backlog of stories. This could also reflect rework that was being identified and resolved. Current percent complete does not indicate a need for a re-plan but progress and velocity should be watched to ensure the team can complete the remaining backlog over the next two minor iterations.

Figure 7 shows a product level view of completion, for Features Committed. Monthly data is graphed, along with a cumulative percentage complete.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

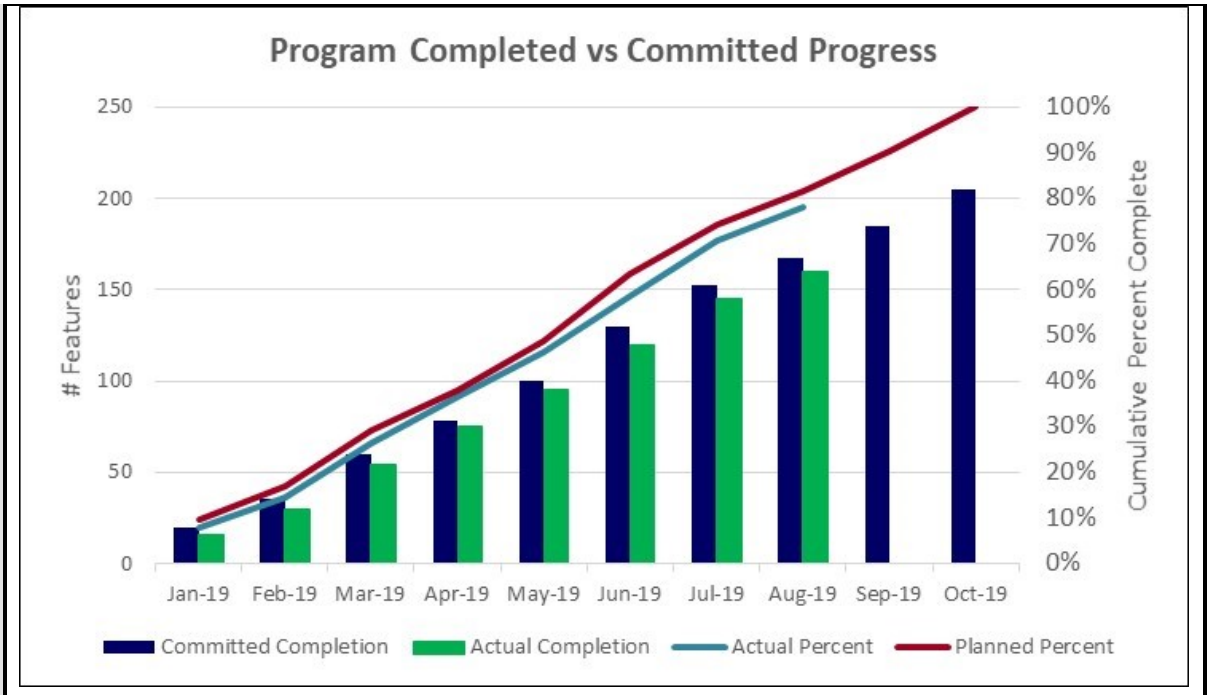


Figure 7: Program Completed versus Committed

The month-to-month cumulative view of the same data shows the project is not completing committed features creating a backlog of work. The gap between planned percent complete and actual percent complete is increasing slightly and is behind target to complete all features by project end. Some corrective actions may be needed.

Analysis Model	Is the team and project completing the assigned work? Will they deliver required features within allocated project schedule? Teams may not complete all Stories for each iteration, so this indicator provides information about any backlog of features growing as you progress through the release or program.
Decision Criteria	If the gap between committed and actual completion is more than 5%, than the team should investigate causes of lack of completions. If any team is more than 10% behind commitments, than the project management should investigate and consider corrective action. If the product completion is more than 10% behind commitments, than alternative courses of action (e.g., adding additional teams or changing commitments) should be considered.

Additional Information	
Additional Analysis Guidance	Use this with the Committed Backlog, Burndown, and Velocity to ensure project will release identified features (or capabilities) as scheduled. The project may want to use different levels of aggregation to view the progress at different levels to expose any adverse trends. If a story is not completed within its expected iteration, it will be placed back on the backlog and re-prioritized for a future iteration. If a team completes assigned stories for an iteration with additional time to work, they should select additional stories from the backlog. Stories, Features, or Capabilities may be weighted by complexity to give a more complete view of program completion.
Implementation Considerations	In general, Committed vs Completed Stories is specific to a team since story point size may vary from team to team. An aggregate measure at the Feature or Capability level can be compiled across teams and compared to capability roadmap to see if project is completing multi-team capabilities within project expectations.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Additional Specification Information	
Information Category	Schedule and Progress
Measurable Concept	Work Unit Progress
Relevant Entities	Stories, Features, or Capabilities
Attributes	Story Points (estimated size), Iteration Committed, Iteration Completed for each entity
Data Collection Procedure	For team measure, data is collected at the end of each iteration by the team lead from the team tracking tool. Story Points must be tested and satisfy "Done" criteria, with no open defects to be counted as completed. If a Story does not satisfy "Done" criteria, then it is not considered "Complete" and its Story Points are not included in the total of Completed Story Points. For product or enterprise measures, data is collected periodically (e.g., monthly, quarterly, end of each iteration or release).
Data Analysis Procedure	Data is analyzed at the end of each iteration by the team during the iteration review and considered during the planning session for the follow-on iteration. The data is also aggregated and analyzed at summary levels across iterations or releases to ensure the program is completing its committed capabilities.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

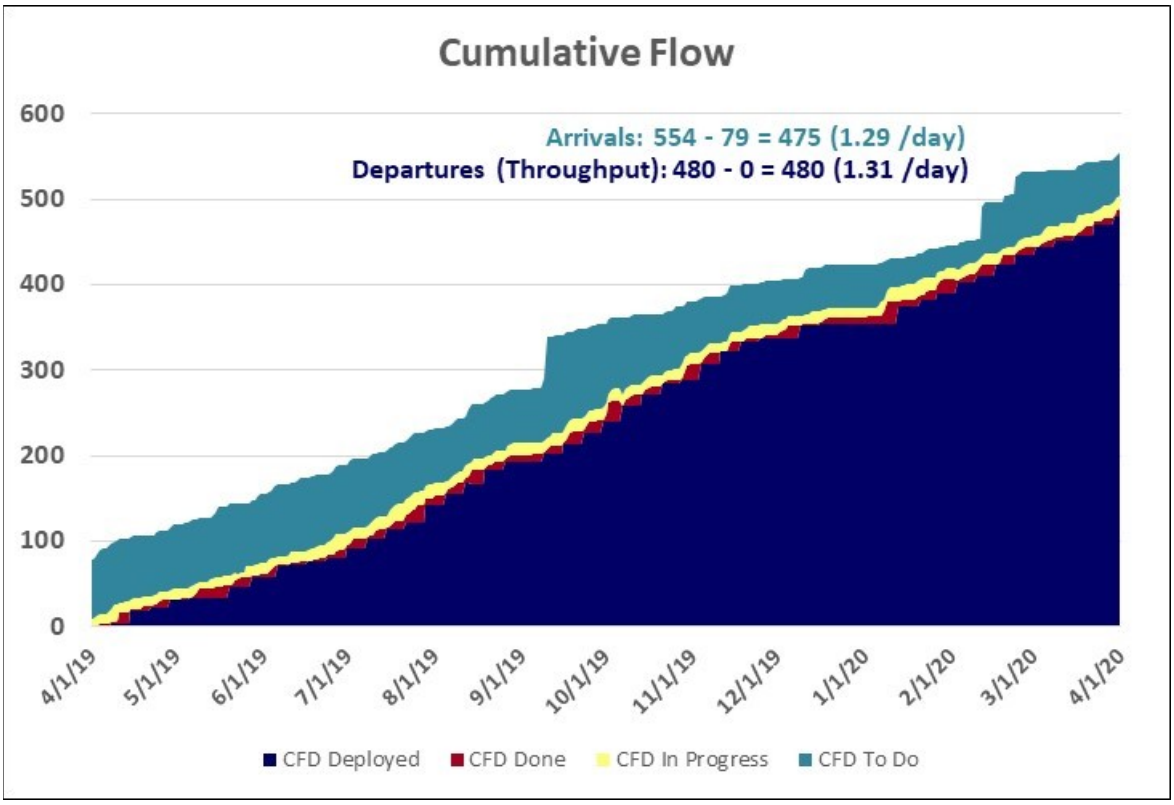
8.4 Cumulative Flow (Team, Product, or Enterprise Measure)

Measure Introduction	
Description	Cumulative flow is a tool to visualize work in progress, cycle time and throughput. In this specification, the indicator (Cumulative Flow Diagram) is described, with base and derived measures that duplicate other measures listed above. Continuous iterative development (CID) methods are focused on the delivery of capabilities/features achieved by managing the flow and throughput of work through a process. Understanding and managing flow is fundamental to achieving stable processes with predictable performance and the efficient use of resources. Flow is visualized and represented graphically in a Cumulative Flow Diagram (CFD) depicting the total quantity and transition of work items in each workflow state over a time period. It is generally desirable that the amount of work distributed across each process workflow state is in balance (new work is equivalent to the completion of work in each workflow state). This can be visualized on a CFD as roughly parallel upper and lower bounds of the cumulative work through each state. Failure to match departures and arrivals for each state can result in queues, backlogs, or inefficiencies in the progress of work completion or utilization of resources. Adherence to effective processes ensuring standard CFD assumptions, rules, and constraints, can help teams achieve predictable performance. Reference: Actionable Agile Metrics for Predictability (Vacanti, 2015)
Relevant Terminology	Cumulative Flow Diagram — A tool used in queuing theory showing whether the flow of work is consistent; visually points out shortages and bottlenecks. Throughput — The number of work items completed per unit time. Work in Progress (WIP) — The number of work units in progress between workflow steps in a process. Work Items — Item that indicates the type of work and what needs to be done (e.g., tasks, stories, features, capabilities). It may include the target date for completion.

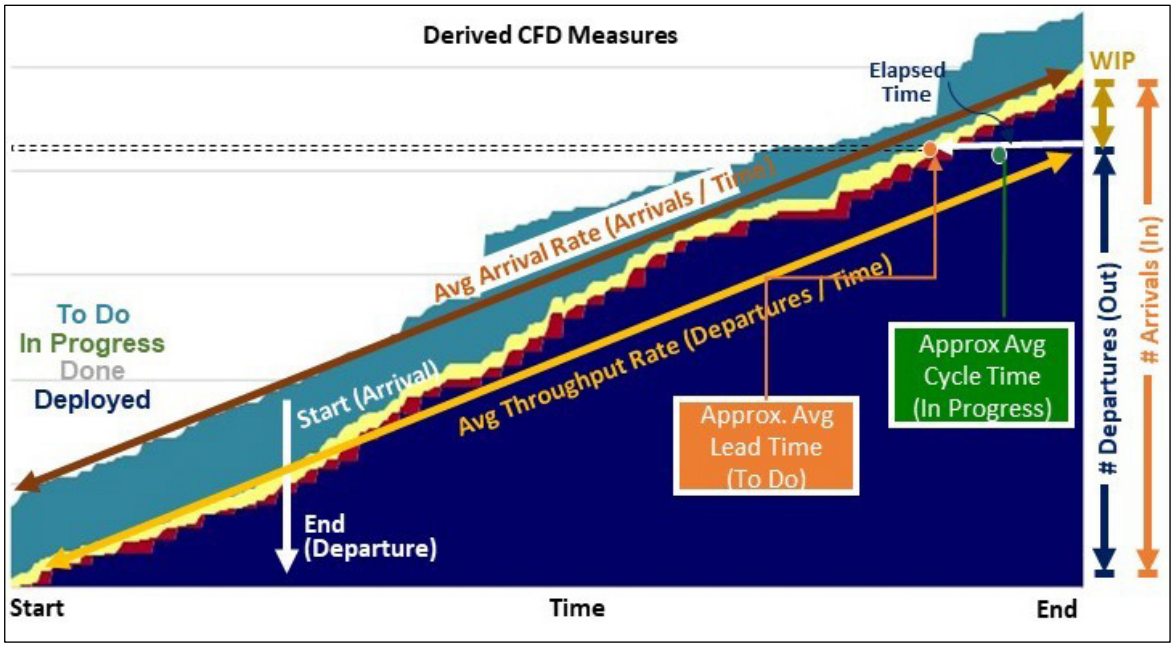
Information Need and Measure Description	
Information Need	Is the flow of work moving forward through the value stream (through the process workflow states)? Is the throughput of work predictable? Are there queues or delays in our process workflows that prevent us from optimizing throughput?
Base Measure 1..N	Base Measures 1-N: The number of work items in each of N workflow states. Collected using counts or times. Note: These states vary by project, organization, or defined process. For the example indicators below, the workflow states used include: • To Do: Work items from the product backlog that have been approved/accepted for implementation (committed to), but not yet started. They generally have been assigned to an iteration or release. The product backlog may also include items that are never implemented. To best depict flow, CFDs do not typically include Backlog work items. • In Progress: Work items that have been approved/accepted for implementation (committed to) and have started development. • Done: Work items have completed all development activities in an iteration and are ready for internal release. • Deployed: Work items have completed all development activities defined by the process, including integration and test activities, and are deployed in an internal or external release.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Derived Measure 1	Approximate Average Cycle Time = average duration for all completed work items Note: The duration is an approximate based on the set of completed work items for a given time range. It is not based on an average of individual work item durations. See Cycle Time / Lead Time specification for a measure based on individual work item durations. - Other derived measures for transitions between workflow states can be calculated similarly.
Derived Measure 2	Throughput = average of Work Items Done per unit time
Derived Measure 3	Work in Progress = average of Work Items in Progress per unit time

Indicator Specification	
Indicator Description and Sample	Flow is commonly depicted in a Cumulative Flow Diagram (CFD), Figure 8, depicting the stacked cumulative quantity of process arrivals, departures, and WIP in bands for process workflow states over time, as illustrated in the example below. The amplitude of the CFD chart indicates the amount of work in each workflow state.  **Figure 8: Cumulative Flow Diagram** This example CFD indicates a project workflow with a team capacity that is well balanced with demand. The number of tasks in each workflow state (height of the bands, or vertical distance between lines) is holding fairly steady and narrow, with relatively parallel lines (slopes) indicating a balance of work arrivals (added to the top teal, To Do, Band) transitioning smoothly into subsequent workflow states culminating in the bottom dark blue, Deployed, band. There are no notable queues, delays, or backlogs (widening CFD bands), except for the arrival of new needs and objectives from the customer in September and March. These are reflected in the Release Backlog (increases in the height of the teal To Do band). These were steadily worked off and implemented by the project team at its consistent rate and capacity (indicated by maintaining fairly stable slopes of the In Progress, Done, and Deployed lines). Throughput rate is steady with no significant changes, except for a short flattening of the progress curves over the December holiday period, that resumed quickly when the team returned to full staffing in January.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	This workflow balance over the year shown is substantiated further by an average task departure rate (1.31 tasks/day), well matched to demand reflected in the average arrival rate (1.29 tasks/day).
Indicator Description and Sample (cont.)	For projects adhering to standards for collection and reporting of CFD data, derived measures for average WIP, average Throughput, and approximate average Cycle Time are related by Little's Law (as discussed in *Actionable Agile Metrics for Predictability*). Generally, these summary cumulative measures can be derived and visualized for a given time range from a CFD diagram as in the abstraction shown in Figure 9. The figure below further illustrates these relationships. **Little's Law:** CT = WIP / TH TH = WIP / CT WIP = CT * TH Avg WIP: Tasks in Work (cum) / duration Avg Throughput: Tasks Done (cum) / dur Avg Cycle Time: WIP / Throughput (dur. / task)  **Figure 9: Notional CFD Diagram** Continuing from the above project CFD example, the project average WIP, average Throughput, and approximate average Cycle Time can be calculated and plotted over time, as in Figure 10.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

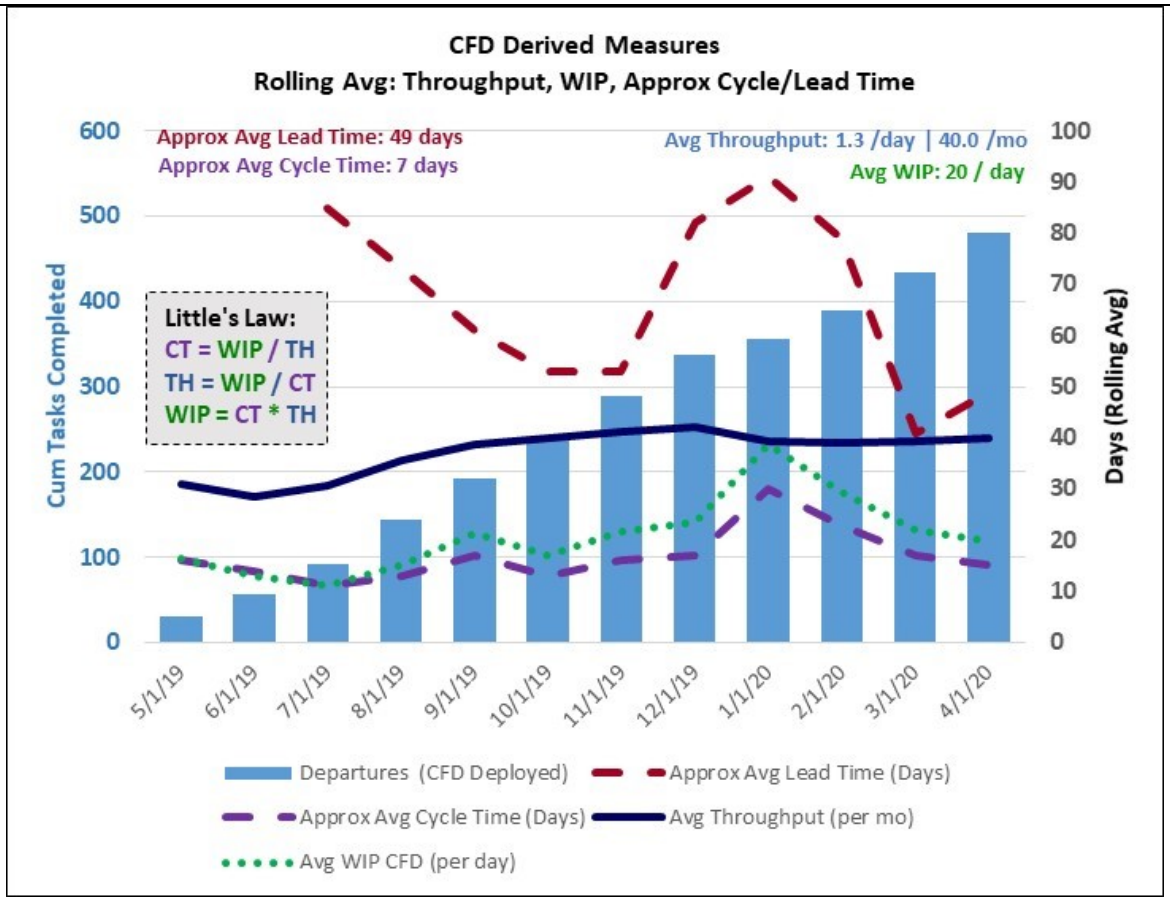


Figure 10: Workflow by Period and Rolling Average

This example provides further numeric substantiation of process effectiveness consistent with the CFD indicator analysis. Derived CFD measures for average WIP, average throughput, and average cycle time indicate fairly stable performance over time that could be useful in predictably planning future estimates. Approximate Lead Time (turnaround for implementing and deploying accepted customer requests) has reduced on average over the last year, even considering the two significant spikes in receipt of new requests and the short delays in throughput over the December holidays.

Note that although CFD measures may indicate stable and consistent workflow process performance, this does not necessarily imply this level of performance fulfills the business need. Process improvements and performance efficiencies may yet be needed to meet the Voice of the Customer. Also note these measures may be specific to the team (e.g., methods for defining tasks, stories, story points) or application domain (e.g., embedded firmware, command and control, information systems, high reliability space applications), so organizations should be cautious about projecting performance across other projects. It may be most beneficial to monitor overall workflow trends and potential areas of concern rather than focusing on absolute measures.

Analysis Model

Is work arriving and being completed at consistent rates? Is there a steady proportionate ratio of WIP across workflow states, or are there queues, delays or inefficiencies indicated by widening CFD bands that should be addressed?

The shapes of CFD bands indicate if the flow of work is being processed and completed at predictable steady rates (e.g., consistent slopes with relatively parallel bands). Other shapes (e.g., diverging bands, flat lines, S-curves) can indicate inefficiencies, mismatched arrivals and departures, or delays in completing the flow of work.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

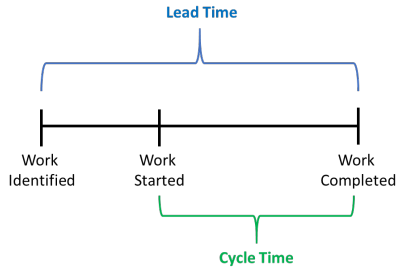
	Is cycle time and throughput compatible with achieving the project plan and product roadmap? Are these measures stable? Comparing derived average cycle time against actual calculations (see Cycle Time/Lead Time specification) can indicate potential process anomalies, such as giving preferential priority to certain tasks. What can be done to increase throughput or reduce WIP, if necessary, to meet performance objectives? Additional details of CFD derived measures and related topics such as technical debt are beyond the scope of this specification and are described further in referenced materials.
Decision Criteria	Significant variations (e.g., ± 10%) in the slope or width of CFD workflow band curves may indicate performance issues, queues or delays in bringing work to closure. Root causes should be analyzed, and corrective actions implemented as appropriate to bring workflow back within expected ranges needed to execute the plan.

Additional Information	
Additional Analysis Guidance	Anomalous CFD band shapes indicating potential delays or negative trends in WIP, cycle time, or throughput may require analysis of root causes. Often reducing WIP or batch sizes can improve process throughput and stability.
Implementation Considerations	CFDs are often available as built-in reports from common agile workflow management tools, which provide additional filtering and reporting options according to the process workflow states in use. CFDs can also be constructed based on measures collected, analyzed and reported using spreadsheet tools. The sample intervals for collection or analysis of CFD data items (e.g., daily, weekly, monthly) may vary based on the program's defined processes or business environment.

Additional Specification Information	
Information Category	1. Schedule and Progress 2. Process Performance
Measurable Concept	1. Work Unit Progress 2. Process Effectiveness
Relevant Entities	Tasks, stories, features, capabilities.
Attributes	Arrivals / departures for workflow state transitions
Data Collection Procedure	Workflow state information (quantities by state over time) and Cumulative Flow Diagrams are typically obtainable directly from software task planning and management tools.
Data Analysis Procedure	Cumulative flow is analyzed by the team regularly (e.g., daily or weekly) to monitor work in progress and completion. Measures are analyzed periodically (e.g., monthly, quarterly, end of each iteration or release) to determine if process performance levels are in line with objectives and sufficient to meet work remaining in the project plan. Corrective actions and process improvements are identified to bring performance within expectations as needed.

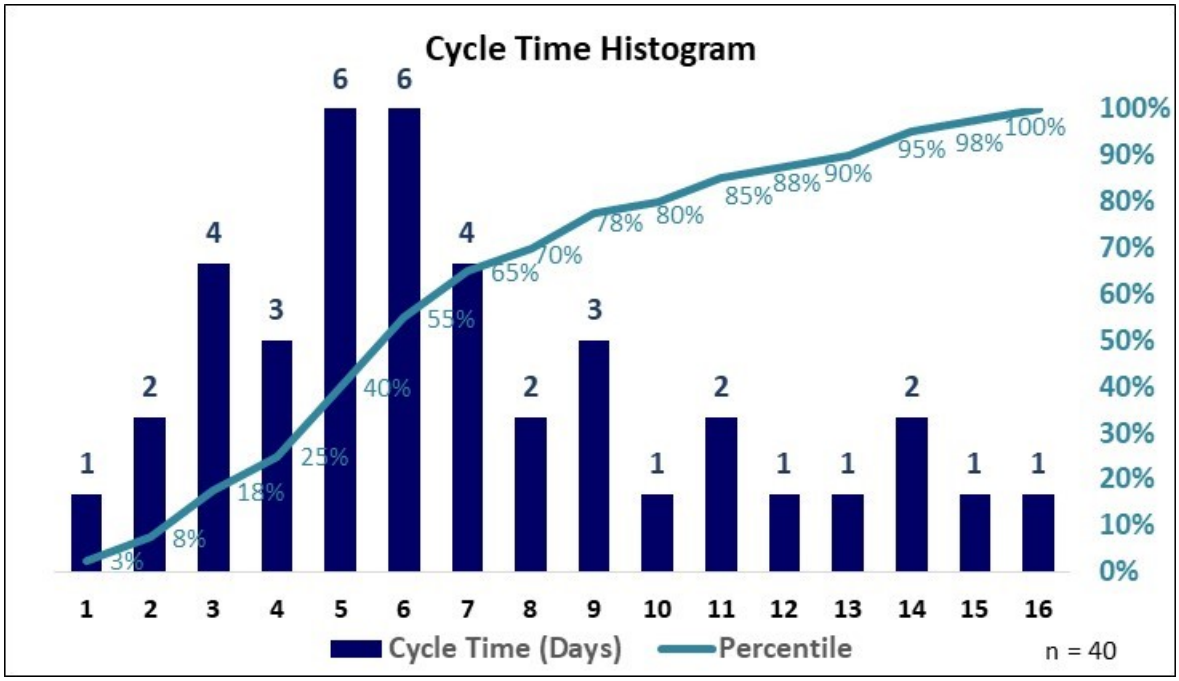
Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.5 Cycle Time/ Lead Time (Team or Product Measure)

Measure Introduction	
Description	Cycle Time and Lead Time can be used to evaluate efficiency in developing work products and as predictors for estimating future work. Cycle Time and Lead Time are similar and related measures that determine the duration for completing new work or products. The differences are in when start times are measured, as depicted in the diagram to the right, and described further below. Refer also to Figure 2, Measurement Context Diagram. 
Relevant Terminology	Cycle Time: The elapsed time from when work is started until the time work has been completed. (e.g., Capability, Feature, Story, Defect). Cycle Time is expressed in terms and context of the team capability. It is typically targeted at measuring repeatability and predictability of team performance for well-scoped work so that results are comparable across multiple similar efforts (stories, features, capabilities). It often excludes the up-front effort needed to define and prepare the work to be implemented, such as backlog, prioritization, planning, requirements analysis, design. Lead Time: Similar to cycle time but is expressed in terms and context of the user or stakeholder perspective. It is measured from the time work is identified and a request is provided to the time until the time it is satisfied. Lead Time includes these up-front necessary activities such as backlog, prioritization, planning, requirements analysis, and design. Lead Time, Cycle Time (and Release Frequency) are closely related measures calculated similarly. The primary difference is in the information need and objective (repeatable team performance vs. user/stakeholder need) which can drive when the start/end times are measured for various activities. Lead Time may also be used to measure a higher-level aggregate business need, as opposed to Cycle Time which may measure the base elements needed to ultimately satisfy that business need.

Information Need and Measure Description	
Information Need (Cycle Time)	How long does it take to release a viable product *(team, product, enterprise)*
Information Need (Lead Time)	How long does it take to deploy an identified feature/capability, once a request is submitted? *(product)*
Base Measure 1	Start time for a process activity *(date and time)*
Base Measure 2	End time for a process activity *(date and time)*
Derived Measure 1	Elapsed Time = (End Time – Start Time) + 1 *(Units may vary based on team context, capability, cadence; e.g., hours, days, weeks, months. May also vary based on calendar time vs. work days. Results with fractional values are rounded up to the next unit.)* Examples: 1: Cycle Time = 08/21/2019 – 08/20/2019 = 2 days 2: Cycle Time = Fri 09/13/19 – Mon 09/02/19 = 12 calendar days = 10 workdays = 2 work weeks 3. Cycle Time = 09/01/19 12:52 – 09/01/19 08:05 = 5 hours 4. Lead Time = 08/31/19 – 6/15/19 = 78 calendar days

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Indicator Specification	
Indicator Description and Sample	 **Figure 11: Cycle Time: Closed Issues** Cycle time performance is frequently analyzed in histograms as depicted in Figure 11. In this example, work items complete in 4 days or less only 25% of the time; 80% of work items complete in 10 days or less; 90% of work items complete in 13 days or less. Analyses such as this can be used to define and monitor process performance objectives, such as service level agreements. Other tools and charts are also common in industry, but typically include information such as: • Plots of cycle time or lead time measures for software deliveries over a defined time range. • Statistical analysis of process performance measures (e.g., mean, median, rolling average, standard deviation)
Analysis Model	Analysis of Cycle Time or Lead Time measures can indicate process performance trends or potential indicators of issues for root cause analysis and performance improvement. Example analyses may include: • Process efficiency and stability (increase/decreasing delivery times or throughput) • Predictability for future performance (narrowing or widening standard deviation in delivery outcomes) The analyst may consider questions such as: • Is the cycle time consistent across iterations? • Is cycle time increasing or decreasing? • Do the cycle time and lead time performance (Voice of the Process) meet the business need (Voice of Customer)? • How predictable is the release cycle? Can we reliably estimate future performance? • What are the root causes for process outliers? • Are process improvements effective? • Are any corrective actions needed to bring performance in line with expectations? Shorter cycle times can indicate effective delivery flow and quicker time to market. Longer cycle times are often correlated to the number of items for Work in Progress (WIP). Consider moderating attributes of the

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	assigned work and resources in order to achieve predictable performance. Tuning small batch sizes for WIP is a common approach used to achieve a consistent delivery cadence. Teams should implement improvements to bring capability and performance in alignment with the business need. Lead times and release frequency can be optimized by managing backlog depth to reduce latency of critical capabilities or applying additional resources to work concurrently.
Decision Criteria	Investigate outliers for cause of variations. Review each outlier that is more than 10% from the average cycle time.

Additional Information	
Additional Analysis Guidance	Under consistent conditions, cycle time and lead time can be used as measures of team capability and throughput that can be used in lieu of traditional size-based productivity measures (such as lines of code / hour). Reductions in cycle time and lead time measures can indicate faster delivery to the customer, which yields additional potential business benefits such as: • Increased productivity • Identification of innovation opportunities • Higher customer satisfaction and employee satisfaction One might divide cycle time further into development effort time, integration effort time, and deployment effort time, if these activities are allocated to separate teams. This way, the team can analyze their current end-to-end performance and take appropriate action as warranted, e.g., allocate more integration hours.
Implementation Considerations	Cycle time and lead time measures can be automatically collected and analyzed by many common tool suites. Refer to Data Collection Procedure for details.

Additional Specification Information	
Information Category	Process Performance – Process Effectiveness
Measurable Concept	Process Efficiency - Speed
Relevant Entities	Features, Stories; Defects
Attributes	Time stamps for process state transitions (start, end)
Data Collection Procedure	Cycle Time and/or Lead Time indicators are often generated directly from software project management tools. Data for these indicators can also be collected manually from Excel.
Data Analysis Procedure	Data is analyzed at the end of each iteration by the team during the iteration review and considered during the planning session for the follow-on iteration. Performance trends of team or organizational capability may be analyzed at periodic intervals (e.g., quarterly) by the program to assess systemic issues and identify improvement actions to align performance with business objectives.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.6 Defect Detection (Team, Product, or Enterprise Measure)

Measure Introduction	
Description	Programs strive to deliver products of acceptable quality for use by internal or external customers, and to manage the extent of defects and rework that could inhibit the effective use of these products in operations. Acceptable quality can often be a tradeoff against other attributes, such as speed, cost, and time to market. Quality objectives may vary by application domain and the business goals of the enterprise, but the objective is generally to minimize the quantity of defects detected after release (escaped) or conversely, to maximize the defects detected during development prior to product release (contained). This may be accomplished through defect detection processes such as effective peer reviews, automated testing throughout development, and other verification and testing approaches.
Relevant Terminology	Defect terminology is defined in Section 2.3 and Section 3 of Part 1..

Information Need and Measure Description	
Information Need	How many defects were contained (discovered) prior to internal release? How many defects were released (escaped) to an internal customer (e.g., Integration and Test, Formal Test) or released (escaped) to an external customer (e.g., end users)? For each major release, how many defects were detected in internal development (contained, saves)? What is the ratio of escaped defects (internal and external) to all defects? Does committed work (stories, features, capabilities) work as expected?
Base Measure 1	Contained Defects (integer scale)
Base Measure 2	Internally Escaped Defects (integer scale)
Base Measure 3	Externally Escaped Defects (integer scale)
Derived Measure 1	Total Defects = Contained Defects + Internally Escaped Defects + Externally Escaped Defects
Derived Measure 2	Internal Defect Escape Ratio = Internally Escaped Defects / Total Defects
Derived Measure 3	External Defect Escape Ratio = Externally Escaped Defects / Total Defects
Derived Measure 2	Total Defect Escape Ratio = (Internally Escaped Defects + Externally Escaped Defects) / Total Defects
Indicator Description and Sample	The concept of categorizing defects as either contained or escaped is key to this measure and others (e.g., Defect Containment). As shown in Part 1 Section 2.3 Figure 3 and repeated below in Figure 12, all defects detected before the release (during development, noted in the blue box) are Contained Defects. All defects detected after release in internal or external operations (noted in the beige and orange boxes) are Escaped Defects.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

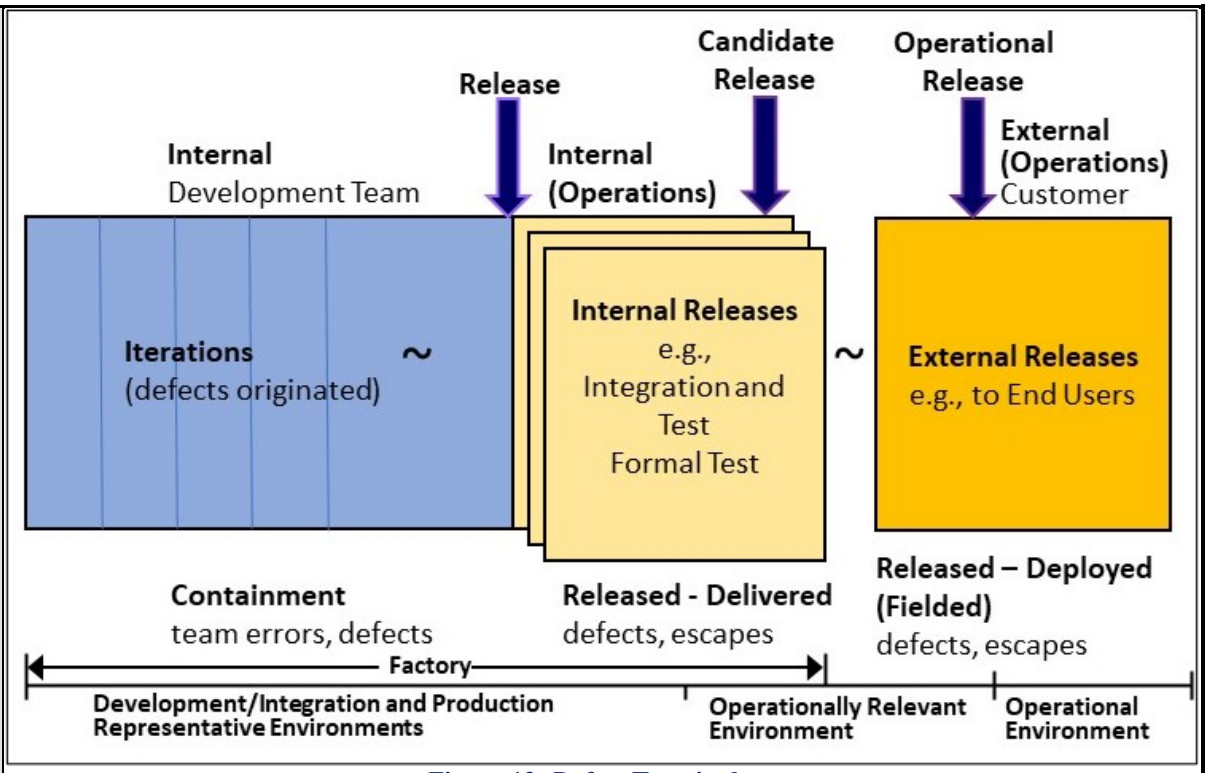


Figure 12: Defect Terminology

The Defect Escapes table (Table 1) is used to show Contained and Escaped Defects for each release along with the Defect Escape ratio. This measures the quality of the completed product based on the number of defects detected before release (Contained Defects) and after release (Escaped Defects). It also monitors the effectiveness of defect detection processes and verification activities performed during development to detected defects prior to release. Note: while only major releases (e.g., 1.0, 2.0, 3.0) are external releases, it is possible to detect external escapes attributed to minor releases after investigation and assignment of iteration introduced.

Table 1: Defect Detection by Release

	Defects				Escape Ratio		
		Escaped					
Release	Contained	Internally Escaped	Externally Escaped	Total Defects	Internal Escape Ratio	External Escape Ratio	Total Escape Ratio
Release 1.0	48	9	3	60	15%	5%	20%
Release 1.1	55	5	1	61	8%	2%	10%
Release 1.2	31	4	0	35	11%	0%	11%
Release 2.0	64	5	2	71	7%	3%	10%
Release 2.1	55	8	0	63	13%	0%	13%
Release 2.2	48	4	0	52	8%	0%	8%
Release 2.3	31	3	0	34	9%	0%	9%
Release 3.0	20	1	0	21	5%	0%	5%
Cumulative	**352**	**39**	**6**	**397**	**10%**	**2%**	**11%**

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

In the example above, Release 1.0 had a ratio of 20% of total escaped defects, with 5% of recorded defects detected after release to the customer. This gradually improved over time to a ratio of 5% on Release 3.0. This was due to a more stable set of requirements, improved test coverage and a more mature product. The Defect Escape Ratio was higher for Release 1.0 because the team decided to implement the more difficult functionality in the first release. Sixty-four defects were discovered in Release 2.0 due to a significant product update. Only 2% of defects were detected externally by the customer.

An alternative way to apply the concept of contained and escaped is to implement the Defect Containment measure. Instead of identifying defects as contained or escaped in relation to the release to an internal or external customer, they would be identified in relationship to iterations. Defects detected in the iteration in which they were inserted (originated) are contained and those detected in later iterations are escaped. Defect counts could be shown in a table as in Table 2 below, identifying which iteration the defects were originated and which iteration the defects were discovered. If this information is unknown, those defects could be tracked separately as Unknown. If legacy defects are detected that were inherited (not originated) by the development team, those could be tracked as Legacy. In a manner similar to the Defect Escape Ratio, various ratios could be determined (e.g., ratio of defects discovered one iteration after they were inserted). See the PSM core framework for more information on Defect Containment.

Table 2: Defect Resolution Lag Time

Defect Resolution Lag Time

As of 19 Dec 19

Defects			(Iteration) 1	2	3	4	5	6		
Defect Discovered (Iteration)	Unknown	0								
	Legacy	0								
	1	82	29	2	19	17	4	11		
	2	123		27	71	6	7	12		
	3	282			122	60	29	71	Blank	0%
	4	112				16	2	94	>1 Iteration	41%
	5	7					5	2	1 Iteration	21%
	6	54						54	Same Iteration	38%
		Total 660	29	29	212	99	47	244		

For this data, 38% of the defects were resolved in the same iteration they were detected. This is less than the organizational goal of 80%. Another 21% were detected in the next release. 41% of defects took at least two iterations to detect, which indicates that the assessment of the iterations needs to be improved, possibly with increased automated test. Some of these escaped defects were not found until after internal release, once an end-to-end test was performed.

Analysis Model

The Defect Escape Ratio is analyzed to determine the quality of a given release and whether the team is improving over time. The Defect Escape Ratio should be getting smaller over time. The defect containment indicator can be used to evaluate the adequacy and completeness of the testing process and the sufficiency of the automated test.

The enterprise may analyze defect escape ratio across multiple programs, especially external escapes, to evaluate those programs that are successfully handling defects.

Decision Criteria

Is the Defect Escape ratio acceptable? Is the ratio getting better over time?

Are at least 80% of defects detected in the iteration where they were originated?

Are at least 98% of defects detected before external release?

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Additional Information	
Additional Analysis Guidance	These tables could be separated by priority (e.g., priorities 1-3 and priority 1) or other attributes. This measure may be used in conjunction with other quality measures including the Defect Density, Defect Resolution, and Rework measures. By looking at both internal and external escapes, the team can determine where improvement actions are needed. A project may intentionally decide to defer defects and add them to the backlog for consideration for resolution in a later iteration or release. These deferred defects may be tagged and tracked separately.
Implementation Considerations	Defects in the problem reporting tool must be discernable whether they were detected before (contained) or after (escaped) the release to an internal or external customer. A parameter or a review of the dates could be used to determine if defects are contained or escaped.

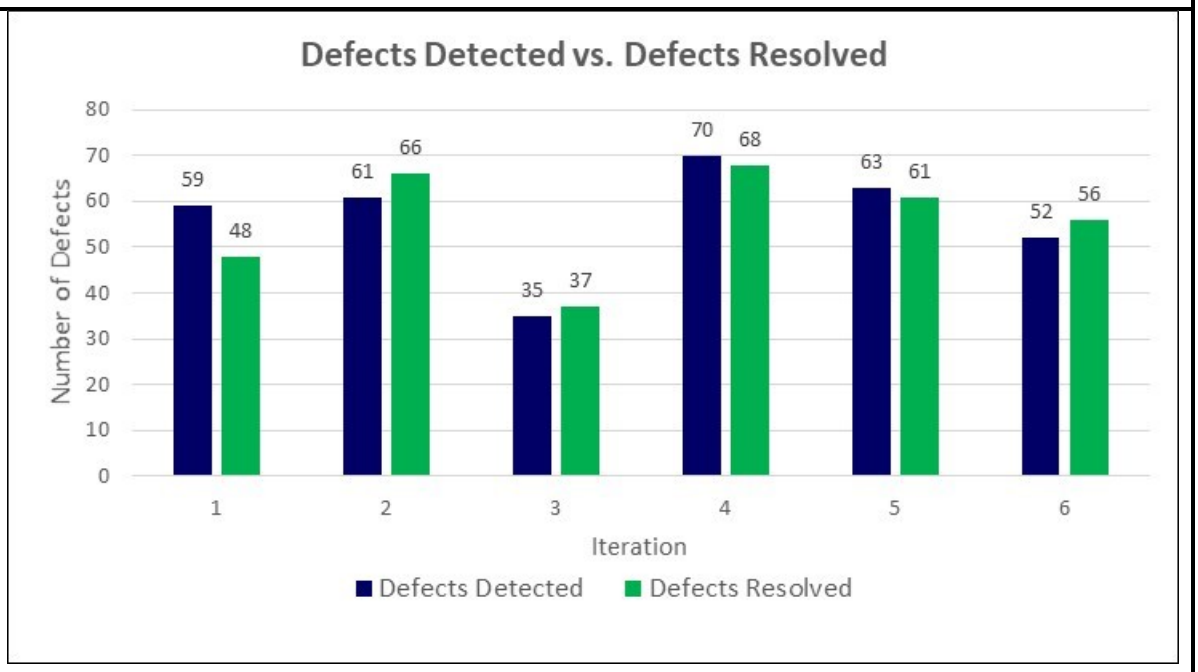
Additional Specification Information	
Information Category	Product Quality
Measurable Concept	Functional Correctness
Relevant Entities	Defects
Attributes	Project activity or iteration where defects are detected (e.g., development, internal release, external release).
Data Collection Procedure	Defect data is recorded in the problem reporting tool as defects are detected. Each defect must be categorized as contained or escaped by assigning a parameter in the tool or by the iteration or date detected.
Data Analysis Procedure	Defect counts and ratios are analyzed at the end of each major release to determine status and progress over time.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.7 Defect Resolution (Team or Product Measure)

Measure Introduction	
Description	Defect Resolution refers to the process of correcting defects that are detected in the system. It is used in conjunction with the Defect Detection measures to ensure that critical defects are resolved in an efficient manner and do not result in inherent quality problems.
Relevant Terminology	The terms defects (team errors), iterations, containment, escapes, and releases is defined in Section 3 of Part 1: Defect Terminology. These terms are also used in the measurement specification for Defect Detection.

Information Need and Measure Description	
Information Need	• When are detected defects resolved? Are high priority defects resolved prior to release? • How many iterations does it take to resolve defects? (aging) • Which defect types have the greatest impact? • Are certain defects taking longer to resolve than others? • How effective was the defect resolution process?
Base Measure 1	Defects detected, per iteration (integer scale)
Base Measure 2	Defects resolved, per iteration (integer scale)
Base Measure 3	Iterations to Resolve (# of iterations between detection and resolution) (integer scale)
Derived Measure 0…n	Resolved 0...n Iteration = the number of defects that are resolved 0..n iterations after being detected Note: Defects resolved in iteration 0, are contained defects.

Indicator Specification	
Indicator Description and Sample	 **Figure 13: Defects Detected versus Resolved** Figure 13 shows that for Iteration 1, not all the defects discovered in Iteration 1 were resolved. These defects were deferred, put on the product backlog, prioritized, and planned to be resolved in upcoming iterations. For Iterations 2 and 3, more defects were resolved than detected, meaning that defects discovered from previous

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

iterations were resolved, thus reducing the product backlog. The gap between Cumulative Defects Detected and Cumulative Defects Resolved creates mission or technical debt, which is added to the product backlog.

Figure 14 shows the cumulative number of defects detected and resolved. In Figure 13 and Figure 14, Iteration 6 was planned to address defects vs. adding new features and capabilities.

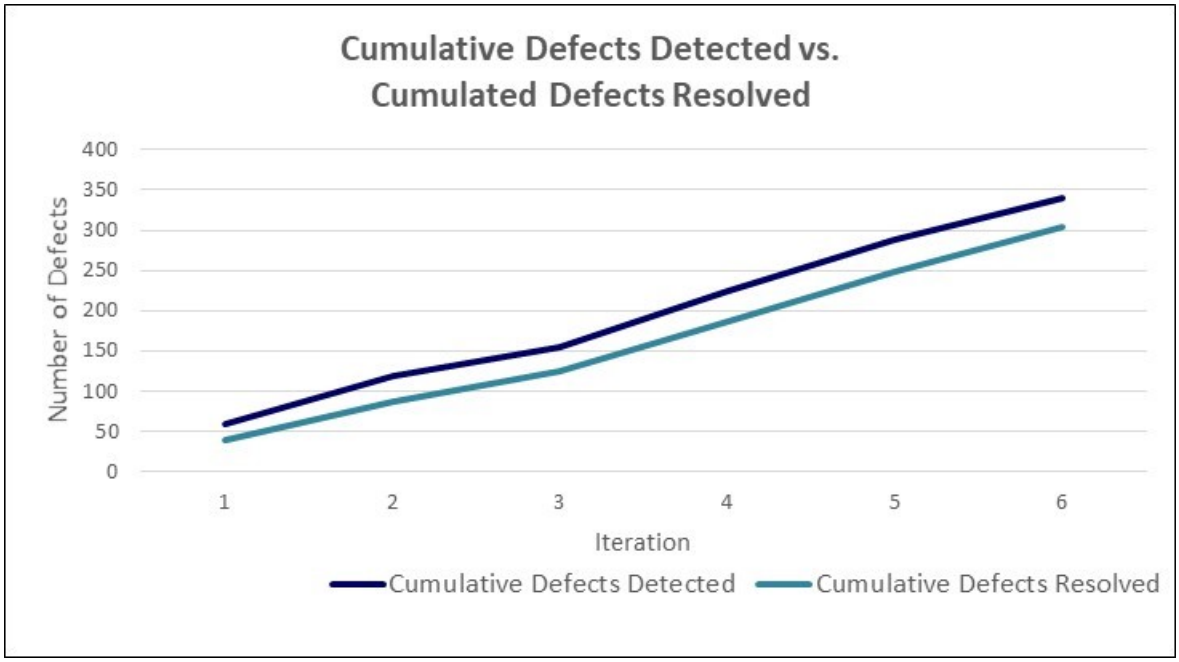


Figure 14: Cumulative Defects Detected vs. Cumulative Defects Resolved

An issue that is often evaluated is how long it takes to resolve discovered defects. In a simplistic case, one can look at how many iterations it takes to resolve the defect. This is shown as a simple bar chart in Figure 15 as Defect Resolution Lag Time. In this example, the defects that took 4 and 5 iterations to fix were lower priority defects dealing with minor changes to screen displays and software documentation.

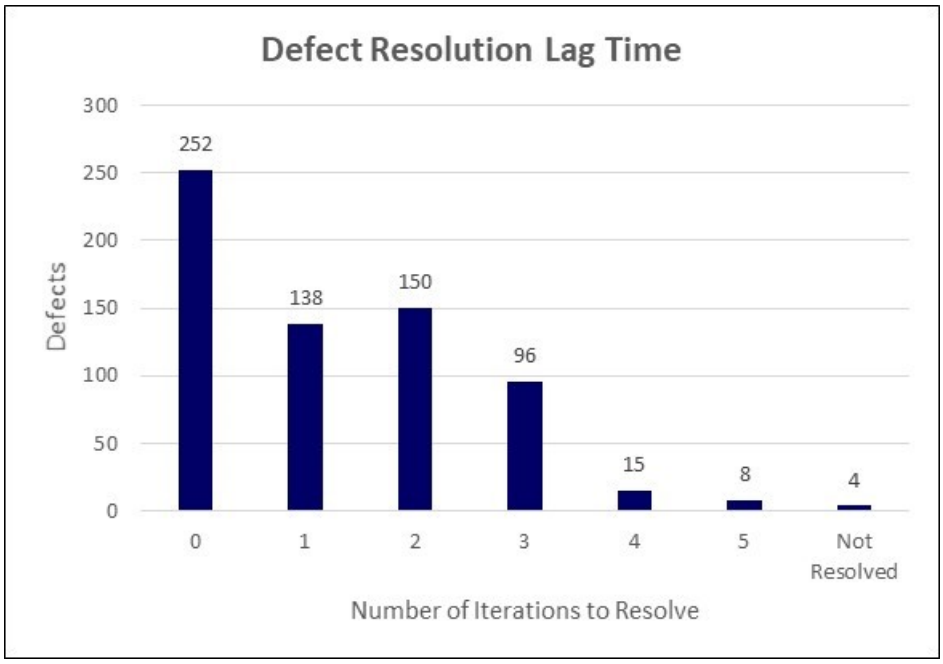


Figure 15: Defect Resolution Lag Time

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Preferably, a defect would be resolved in the same iteration as it was discovered (the teal series of diagonal cells in Table 3 below). All cells to the right of this diagonal represent escaped defects across iterations. Filtering can be applied for the most critical or highest priority defects. Defects that are not resolved after multiple iterations may represent a risk to the inherent quality of the product, may represent an issue with the defect resolution process, or may indicate lower priority defects that have not been prioritized for implementation. Analysis of the Defect Resolution Lag Time measure should focus on the high priority defects and ensure they are being resolved in a timely matter.

Table 3: Defect Resolution Lag Time

Defect Resolution Lag Time

As of 19 Dec 19

			Defect Resolved (Iteration)						
			1	2	3	4	5	6	Not Resolved
Defect Detected (Iteration)	1	59	48	11					
	2	61		55	6				
	3	35			31	4			
	4	70				64	6		
	5	63					55	8	
	6	52						48	4
	Total	340	48	66	37	68	61	56	4

Blank	0%
>1 Iteration	0%
1 Iteration	10%
Same Iteration	89%

Analysis Model

Figure 13, Defects Detected vs. Defects Resolved, shows the difference/delta between defects discovered and defects resolved, by iteration.

The Cumulative Defects Detected vs. Resolved indicator can be used in conjunction with the Feature or Capability Backlog measure. When checked cumulatively, if the number of defects discovered is greater than the number of defects resolved, the backlog is growing. If the number of defects discovered is less than the number of defects resolved, the backlog is getting smaller.

Decision Criteria

In Figure 13, for each defect that does not get resolved in the same iteration as it is discovered, the defect and its priority shall be considered during the planning session for the follow-on iteration.

In Figure 14, when the difference/gap between cumulative defects discovered and cumulative defects resolved exceeds 20% of the cumulative defects discovered, the team shall consider having an iteration specifically designed to resolve the outstanding defects.

In Figure 15 and Table 3, defects with Priority 1 and 2 should have a defect resolution lag time not greater than 1 iteration. If not, the defect shall be considered for resolution in the next iteration, with customer approval of this action. Priority 3 through 5 defects may be deferred until later iterations, based on customer priorities.

In Figure 15 and Table 3, most Priority 1 and 2 defects should be resolved prior to release (e.g., a condition of release). Some may be deferred to a later release, with customer agreement. Priority 3 through 5 defects not resolved may be released with customer approval and have a customer approved work around.

Defect detection and resolution data is often presented and used as a criteria for phase completeness at phase gates and associated reviews.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Additional Information	
Additional Analysis Guidance	Considering the nature of agile development, a defect lower in severity and priority in the product backlog may not be resolved immediately but, be deferred to be resolved in a later iteration. To account for this planned delay, the Defect Resolution Lag Time could be derived from the Iteration the defect was resolved to the Iteration the defect was planned to be resolved (instead of Iteration the defect was detected). The derived measure for Defect Resolution Lag Time listed above is measured for defects that were resolved. The lag time for open, unresolved defects would be calculated by the Current Iteration less the Iteration the defect was detected. Digital Engineering/Model Based Engineering should result in early verification and product specification completeness in earlier lifecycle phases accomplished via models and digital system views. A particular emphasis is determining if defects are both detected and resolved in earlier phases than previous performance, and that defects are resolved as early as possible. More advanced analysis may evaluate (new) defect insertions during defect resolution, or defects resolutions that failed. Recurring rates may be an important customer concern.

Implementation Considerations	Counting methods need to be defined to determine:
	• What constitutes/does not constitute a defect
	  • E.g., peer review findings may be considered errors and not considered internal defects
	  • E.g., an internal error that is sent back to the originating team and results in rework, may be considered a defect
	• When defects will/will not be counted (e.g., upon hand-off to another team/3rd party)
	• Internal defects vs. external defects (e.g., defects discovered by the developer, by the customer in an operationally representative environment, or by the customer in operations)
	Determining a value for the Iteration the defect was detected and the Iteration the defect was resolved may be tool dependent.
	As an alternative view, these measures and indicators may be constructed using only Priority 1-3 defects that affect functional performance.
	Some iterations may consist of only defects resolutions. Keep this contextual information in mind when it comes to analyzing the data.

Additional Specification Information	
Information Category	Product Quality
Measurable Concept	Functional Correctness
Relevant Entities	Defects
Attributes	Iteration Defect was Detected Iteration Defect was Resolved Defect Priority
Data Collection Procedure	Data is collected at the end of each iteration by the team lead from the team tracking tool.
Data Analysis Procedure	Iteration the defect was detected and Iteration the defect was resolved are discussed during the defect tracking and defect resolution meetings. Data is analyzed at the end of each iteration by the team during the iteration retrospective meeting and considered during the planning session for the follow-on iteration.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.8 Mean Time to Restore (MTTR) / Mean Time to Detect (MTTD) (Product or Enterprise Measure)

Measure Introduction		
Description	In an operational environment, continuity of deployed services is fundamental to the delivery of user value. MTTR is essential for systems in which operational availability is critical. This includes both critical embedded systems as well as those systems focused on the delivery of software services. Operations can be impacted by planned or unplanned outages. Operational service incidents are typically recorded in a trouble ticket which is used to track the incident to closure and restoration of service. Each trouble ticket has an associated restoration time. Sometimes there may be an alternative or workaround that enables the service to continue in the field, such as redundant paths or resources, even if in a degraded mode. Some repairs must be returned to the factory for correction and redeployment. The enterprise may collect the average time to detect a service-impacting issue (Mean Time to Detect) and the average restoration time (Mean Time to Restore). This provides measures of operational effectiveness for maintaining service continuity, across all tickets, or classes of tickets. A summary of these concepts is depicted visually in Figure 2, Measurement Context Diagram. MTTR, MTTD and other operational measures of service continuity can be applied in each of many potential stakeholder environments including the development/integration environment(s), production representative environment, or operationally relevant environment, or the operational environment. The enterprise generally focuses on actual measures from the operational environment. The product team may also focus on ensuring MTTR/MTTD objectives will be met as the system is developed and sustained.	
Relevant Terminology	Mean Time to Detect (MTTD)	Time required to identify an interruption to service delivery. MTTD measures how long it takes the operations team to detect that an incident has occurred which affects delivery of operational services.
	Mean Time to Restore (MTTR)	Time required to restore service after an outage occurs. MTTR measures how long it takes the operations team to restore the system to an operational state, either through a rollback, restart, fix in operations, return to the factory for repair, or another action. Sometimes synonymous with Mean Time to Recover, but with a focus on restoration of operations.

Information Need and Measure Description	
Information Need	What is the reliability and availability of operational capabilities? How long does it generally take to restore service when a service incident occurs? How quickly can we recover from failures that impact the system in operations (e.g., impacts service reliability or availability), or the software in development or test? *(time to restore the build or the service to a previous, known good state.)*
Base Measure 1	Failure Occurrence Time *(timestamp)*
Base Measure 2	Failure Detection Time *(timestamp)*
Base Measure 3	Service Restoration Time *(timestamp)*
Derived Measure 1	Time to Detect = (Failure Detection Time) – (Failure Occurrence Time) *(units for elapsed time may vary; seconds, minutes, hours, days)*
Derived Measure 2	MTTD = ∑ (Time to Detect) / N *(rolling average Time to Detect, based on N previous failures)*
Derived Measure 3	Time to Restore = (Service Restoration Time) – (Failure Occurrence Time) *(units for elapsed time may vary; seconds, minutes, hours, days)*
Derived Measure 4	MTTR = ∑(Time to Restore) / N *(rolling average Time to Restore, based on N previous failures)*

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Indicator Specification	
Indicator Description and Sample	When practicing CID, a key concern is speed: to deliver software rapidly and frequently. However, quality should be maintained. In particular, when practicing Continuous Deployment into operations it is important to be able to quickly recover when a new release/deployment introduces a failure in this live environment. MTTD and MTTR indicators can be represented in multiple ways (e.g., graphical, tabular). In Figure 16, three measures are plotted for each operational outage: Time to Detect, Time to Repair, and Time to Restore (sum of detection + repair). A comparison of data across outages indicates general trends, severity, and operational impacts. A summary of statistical measures (mean, median, standard deviation) for each of detection time, repair time, and total restoration time is provided in the table below the chart. A rolling average of Mean Time to Restore (MTTR) is also plotted for the 10 most recent outages.

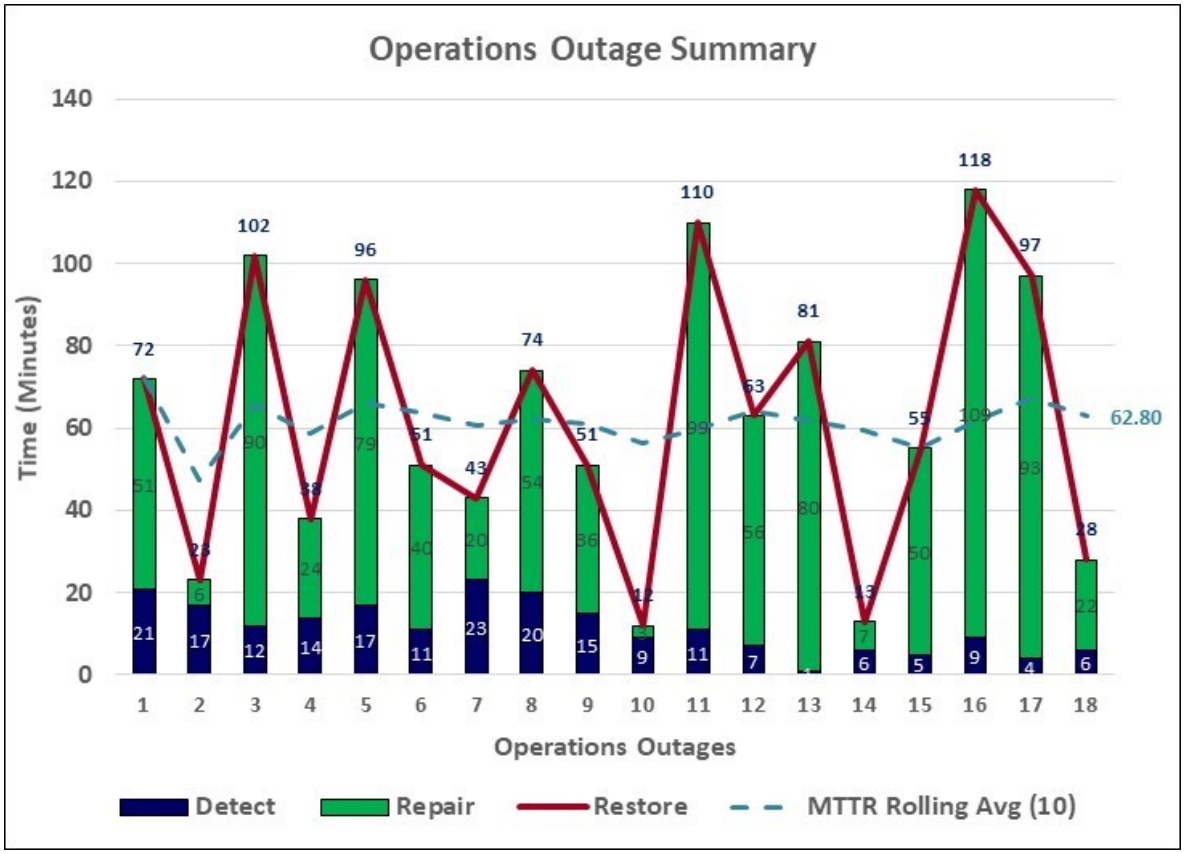


Figure 16: Operations Outage Summary

In this example, although there are significant variations in individual outage samples (some anomalies are more complicated to fix than others), in aggregate the MTTR rolling average is holding fairly steady (around 1 hour to restore service). Similarly, the mean and median times for Time to Detect, Time to Repair, and Time to Restore are consistent despite a large standard deviation. (Table 4)

In the sample indicator, the four short MTTRs are cases where the system was rolled back to a previous version. The longest cases are indicative of complex issues that required additional repair time. The lengthy MTTR in Outage 16 involved an update to a critical component. The fix/corrective action was not implemented correctly, which resulted in Outage 17. An alternative solution was implemented, and the software was shown to work in the next iteration.

Table 4: MTTR Statistics

	Detect	Repair	Restore
Mean	11.56	51.06	62.61
Median	11	51	59
Std Dev	6.31	34.09	33.28

In this example, feedback from the user community indicates outages of greater than 30 minutes can have a significant impact on Operations, due to reports that are due twice hourly. Missing two consecutive reports

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

<table>
<tr><td></td><td>impacts decision making. This example program is considering ways to shorten restore times, such as implementing automated roll-back capabilities where any new deployment/release that introduces a failure can be rolled back and the previous release rapidly restored. Program personnel are also conducting a Pareto analysis of outage times by defect type to determine which outage types are most costly, so that resources can be prioritized on targeted improvement actions.</td></tr>
<tr><td>Analysis Model</td><td>
Data is gathered from service incident tickets and classified or filtered into affinity groupings of interest (e.g., priority, type, component, severity, impact, duration, detection method). Trends and root causes are evaluated. Improvement plans may be defined and implemented with corrective/preventive actions to mitigate the frequency or impact of future occurrences, as appropriate, relative to business objectives. The effectiveness of improvement actions should also be measured.
Both MTTD and MTTR need to be evaluated as to whether they meet the business/mission needs in terms of reliability and availability. Projections and actuals are evaluated against objectives, and trends are analyzed to project whether required objectives will be met.
A good pipeline should include significant automated testing such that any failure-inducing defects or issues are detected before deploying into the operational environment.
MTTD and MTTR are measures of failure trends for a set of issues across a range of time, and they characterize the capability to maintain and rapidly restore operations and operational service. Analysis and improvement actions can vary based on the situation and trends of performance measures and whether these are reliable predictors of future performance so improvement actions can be effective. Examples of potential areas for investigation are summarized in the table below:
<table>
<tr><td>Trend</td><td>MTTD</td><td>MTTR</td></tr>
<tr><td>Increasing</td><td>• Ineffective monitoring, detection processes, tools, training
• Incomplete knowledge of failure modes</td><td>• Increasing complexity of system, software, or architecture
• Lack of rollback capability or strategy
• Lack of effective redundancy
• Developer changes / inexperience</td></tr>
<tr><td>Steady</td><td>• Established MTTD met and satisfied - no further improvement needed
• Predictable capability; does it meet the business need (Voice of the Customer)?
• Lack of continuous improvement</td><td>• Established MTTR met and satisfied - no further improvement needed
• Predictable capability; does it meet the business need (Voice of the Customer)?
• Lack of continuous improvement</td></tr>
<tr><td>Declining</td><td>• Improved monitoring effectiveness
• Defect prevention initiatives</td><td>• Improvements through automation, tools
• Added capability or capacity (redundancy, etc.)</td></tr>
<tr><td>Erratic</td><td>• Inconsistent monitoring or reporting processes</td><td>• Unstable processes
• Immature system
• Ineffective process improvement</td></tr>
</table>
</td></tr>
<tr><td>Decision Criteria</td><td>
After deployment, when MTTR or MTTD is above mission or business objectives, a decision as to whether the system should be rolled back to a previous version may be considered. If the decision is not to roll-back, the user may create a high priority change request to resolve the issue causing the high MTTR. Increasing trends in MTTR or MTTD measures, may also lead to the creation of new defects or stories to improve performance, or the need to evaluate and improve the development/test processes. This is especially important when a safety critical or mission critical failure occurs.
When additional defects are introduced after improvements are made, special attention should be applied to the resolution process.
During development and test, for any MTTD or MTTR that is more than 10% above the objective or mean, investigate the root cause(s) and decide if additional improvements or testing is required. Trends over time should be improving (getting smaller) as additional functionality is added and as the system nears deployment. Regular occurrences above the objective may mean that the system is not mature enough for operations, and deployment may need to be delayed. For trends that are increasing above the objective or mean, additional focus or process improvements may be required.
</td></tr>
</table>

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Additional Information	
Additional Analysis Guidance	MTTR is an essential measure for systems in which operational availability (Ao) is critical, with a focus on safety-critical and mission-critical failures. MTTR is also paramount when practicing full continuous deployment into Operations: in this case Operations is an operational environment supporting live operations/missions and thus the system must maintain high reliability and availability. However, even in testing environment, a failure means that integration or test activities are impacted (and possibly deployment which may lead to cost/schedule overruns). Additional analyses of MTTR / MTTD measures can be utilized to determine appropriate actions to improve availability and rapid recovery from operational issues. Examples include statistical analysis methods, profiles of defect distribution or characteristics, Pareto charts, root cause analysis, or other quality management tools.
Implementation Considerations	Measuring individual failures and restorations should be automated as much as possible, based on timestamps in logs or other automated data collection mechanism.

Additional Specification Information	
Information Category	Process Performance
Measurable Concept	Process Efficiency – Speed Supportability – Maintainability – Dependability – Reliability
Relevant Entities	Service incidents
Attributes	Time of outage, detection, and restoration; defect priority and reason code; affected elements
Data Collection Procedure	Date/time is collected at the start of each failure or service outage, and at the time of operations or service restoration. The delta between these is the individual outage TTR. These are collected to calculate a historical mean MTTR.
Data Analysis Procedure	Data is analyzed periodically during development and test, and trends are evaluated. During operations, data is analyzed when safety or mission critical failures occur, as well as periodically.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.9 Release (or Deployment) Frequency (Product or Enterprise Measure)

Measure Introduction	
Description	As described in Overarching Principles, products are typically planned and developed iteratively (e.g., capabilities, features, stories, tasks) into a set of internal releases, candidate releases, and deployed product releases. This is represented conceptually in Figure 17.

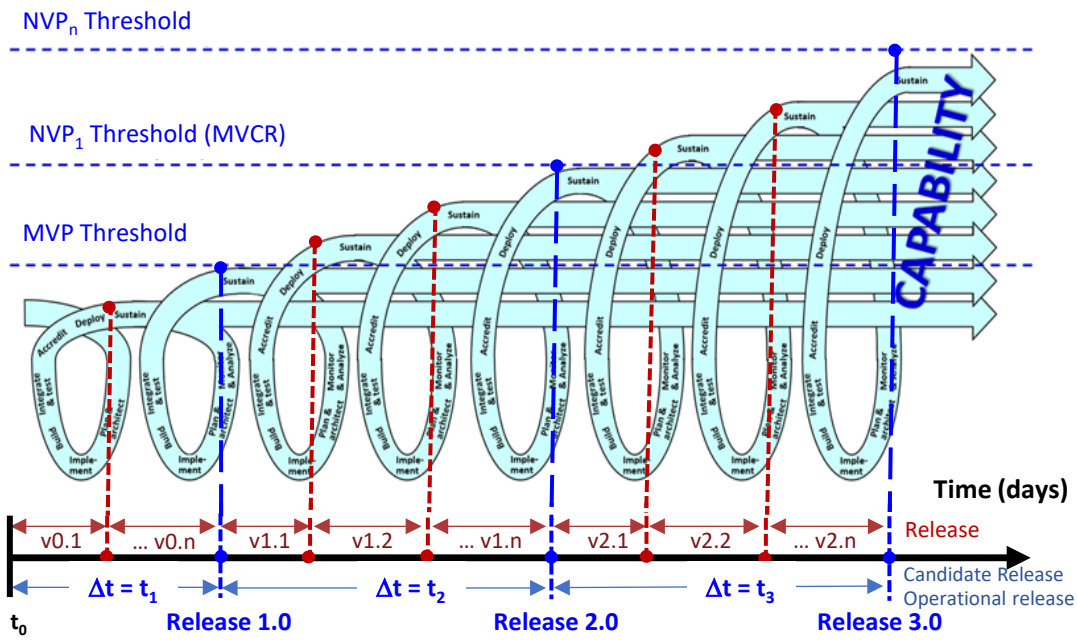


Figure 17: Iterative Development

The speed and frequency at which products are released are crucial in providing useful capability to users as rapidly as possible. The scheduling, duration, and frequency of releases can vary widely (e.g., months, weeks, days, or on demand) based on domain or business need. Products may be iteratively released on a predictable fixed cadence, or on demand as needed. The time and effort to develop candidate product releases and transition them to deployed external product releases are primary measures of efficiency in making features/capabilities available to users, as depicted in Figure 18.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

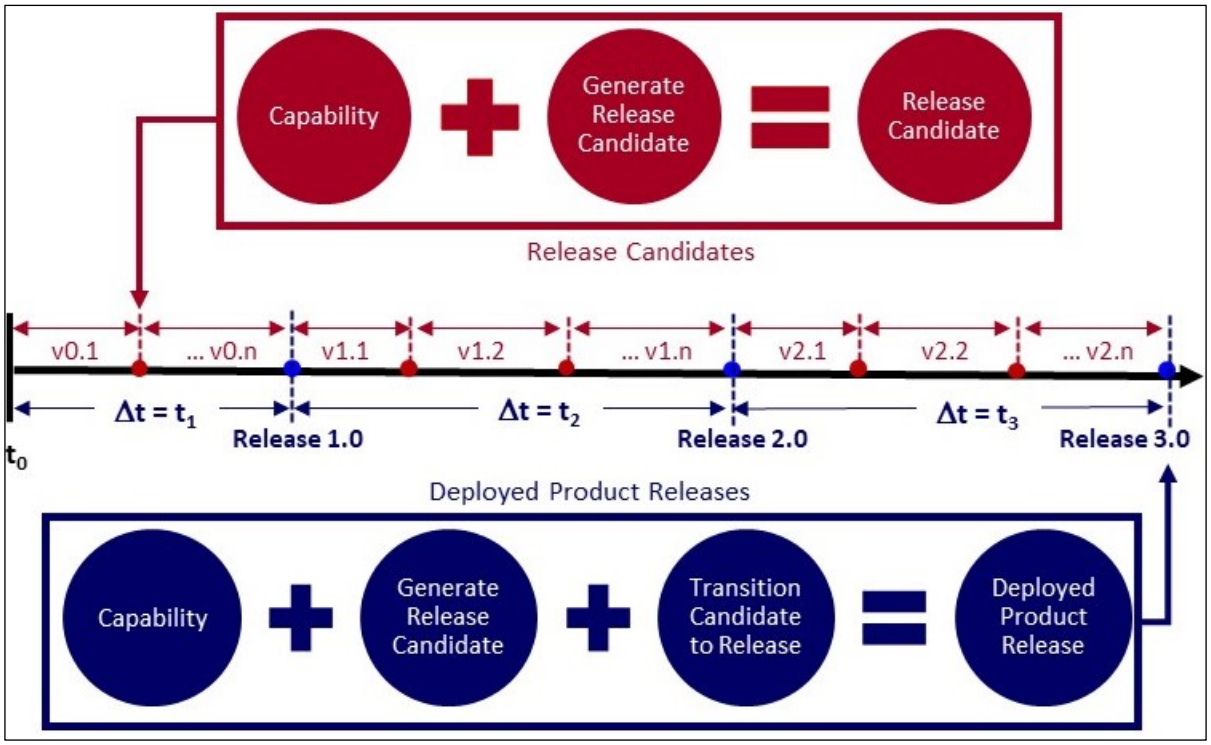


Figure 18: Product Iterative Releases (Conceptual)

Relevant Terminology	MVP — Minimum Viable Product MVCR — Minimum Viable Capability Release NVP — Next Viable Product Release — Internal Release; Candidate Release (External Release); Operational Release (Deployment Release) Refer to glossary for definitions.

Information Need and Measure Description	
Information Need	How long does it take to deploy an identified feature/capability? *[Product]* What is the cadence or frequency for product release or deployment? *[Product, Enterprise]* How long does it take to release a minimum viable product? *[Product, Enterprise]* How much effort/cost/time is needed to develop new products and transition them to release? *[Product, Enterprise]*
Base Measure 1	Release Start Date (datestamp) *(release, candidate release, or operational release)*
Base Measure 2	Release End Date (datestamp) *(release, candidate release, or operational release)*
Base Measure 3	Effort Hours to generate a release (integer) *(internal release candidate or external deployed release)*
Base Measure 4	# of Releases (for a specified data range)

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Derived Measure 1	Release Duration = (Release End Date) – (Release Start Date) Note: release durations may be tracked for features/capabilities at various stages of maturity • Time to Minimal Viable Product (MVP) • Time to Minimal Viable Capability Release (MVCR) • Time to Next Viable Product (NVP_n)
Derived Measure 2	Release Frequency = (# of Releases) / date range (e.g., days, weeks, months, quarters, years)
Derived Measure 3	Average Release Duration = ∑ (Release Duration) / (# of Releases) *Note: weighting can be used to emphasize the most recent releases.*
Derived Measure 4	Average Release Transition Time = ∑ (Release Transition Time) / (# of Releases)

Indicator Specification

Indicator Description and Sample

In this example, (Table 6) a commercial software company deployed a new product (Tango) to the market in October 2018 (MVP release), with a business objective to release iterations twice monthly to support quarterly product capabilities releases. Ten product releases were completed between October 2018 and March 2019. The table below summarizes, for each release, the start and end dates for each release (from which duration is calculated), the type of release, and the total labor spent in hours.

Following the higher effort for the initial MVP R2018.01 release, durations of iterations have averaged 18 days. The initial MVP did not meet market needs, however, a Minimum Marketable Product (MMP) was available two months later in December 2018. After the MMP, the NVP release occurred 90 days later, in line with the business objective of quarterly releases.

A longer duration for the R2019.01 iteration (25 days) at the end of 2018 is attributed to staffing reductions due to holiday vacations. Overall averages for release time and labor across releases is shown in the Table 5, by calendar year.

Table 5: Product Release Averages

Days to Release	2018	2019
# of Releases	5	5
Days to Release (Avg)	25.8	19.4
Labor to Release (Avg)	2402	1671

Table 6: Release Frequency and Labor Hours

Release	Started	Ended	Release Type	Duration	Year	Labor Hours
R2018.01	8/20/2018	10/23/2018	Candidate Release - MVP	64	2018	6182
R2018.02	10/22/2018	11/5/2018	Release - Internal	14	2018	1313
R2018.03	11/5/2018	11/26/2018	Release - Internal	21	2018	1663
R2018.04	11/19/2018	12/5/2018	Candidate Release - MMP	16	2018	1477
R2018.05	12/3/2018	12/17/2018	Release - Internal	14	2018	1372
R2019.01	12/17/2018	1/11/2019	Release - Internal	25	2019	1553
R2019.02	1/11/2019	1/26/2019	Release - Internal	15	2019	1658
R2019.03	1/26/2019	2/11/2019	Release - Internal	16	2019	1389
R2019.04	2/11/2019	3/5/2019	Candidate Release - NVP	22	2019	2002
R2019.05	2/25/2019	3/16/2019	Release - Internal	19	2019	1756

The product team plots each release in the Figure 19 below for a visual comparison of durations (vertical bars aligned with the left axis) and labor hours invested (teal line aligned with the right axis). A rolling average of the durations for the most recent 3 product releases is calculated and displayed in the dashed red line.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

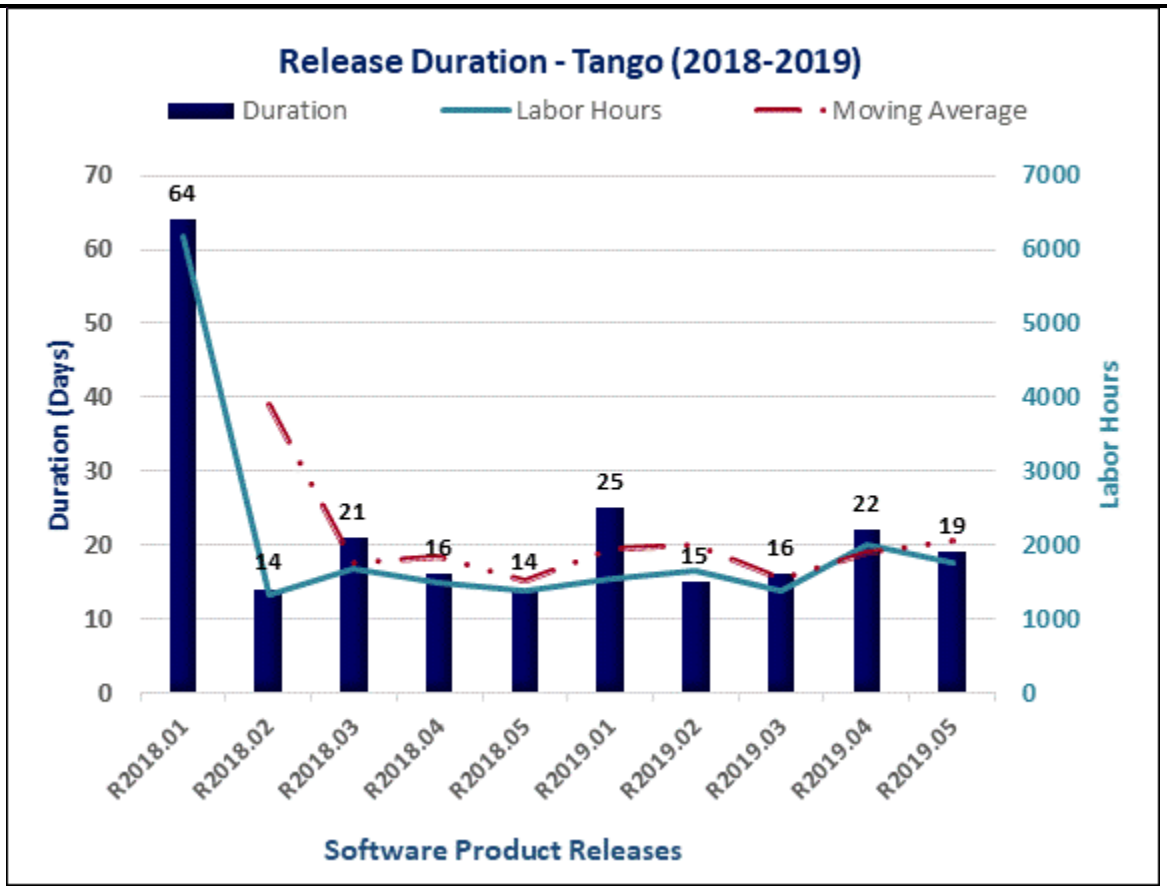


Figure 19: Release Duration for Product Tango

In Figure 20 the marketing department tracks the release frequency for all three of the company's products at the enterprise level against the business plan for twice monthly iterative releases.

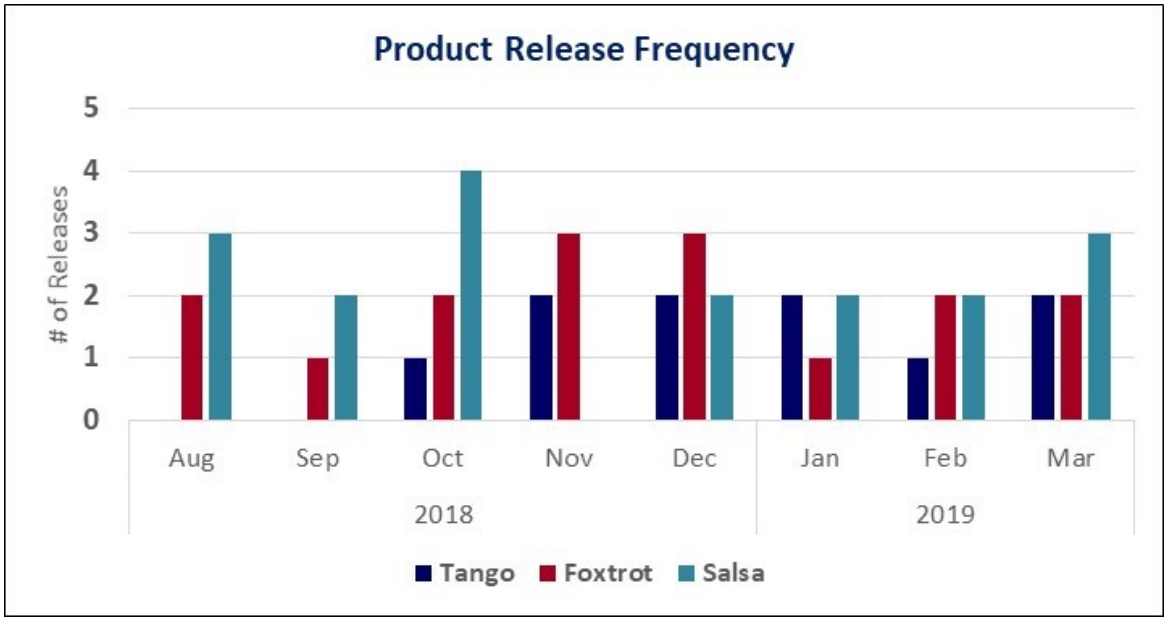
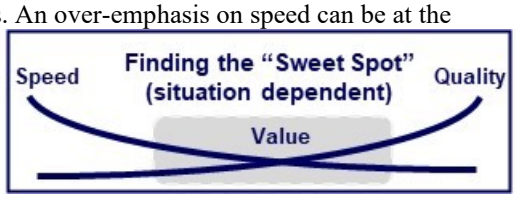


Figure 20: Product Release Frequency

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Analysis Model	Can we consistently release product baselines at a rate needed to meet demand? Is the process performance (time and labor) for generating and deploying product releases improving? Does it take more/less/same amount of time to transition release products to candidate release products or operational release products? Not all development organizations are in control of when their internal baselines may be deployed to live operations. For instance, deployments for military platforms must be certified and coordinated with the user community. As shown in Figure 1, additional effort may be needed to prepare and transition candidate releases to operationally representative environments. This may require a separate set of release measures to manage and optimize the rapid delivery of capabilities to end users. This preparation and transition effort may increase significantly as the baseline grows. Not only must the new capability be verified, so must prior functionality be verified through regression testing. If done manually, the additional effort for testing and release can scale at a rate incompatible with maintaining product release timelines. Automation can help improve build, testing, and release efficiency to maintain a consistent release transition cadence. The time to build and create product releases is directly related to the quantity and size of design and implementation features. Smaller batch sizes enable releasing products more quickly. Efficiency of the deployment and release process further accelerates the speed at which products can be released to the customer. Releases are typically built by automated build/test automation frequently on the baseline. Releases are typically built every day or upon every merge or end of a sprint. The frequency of failures for releases impacts confidence in the software baseline. Ideally over time, releases can be produced more efficiently by replacing manual steps with automation.
Decision Criteria	• If the effort to transition release products to candidate releases or operational releases is increasing steadily beyond performance goals, consider approaches such as automation or reducing batch size to increase release frequency and speed the delivery of capability to users. • Once stabilized, action may be needed if the quality of deployed products declines or if the team is unable to sustain release timelines. • Does adding features/capabilities result in increased cost to create a candidate release or operational release?

Additional Information	
Additional Analysis Guidance	Release frequency (how often?) have close dependencies with Lead Time and Cycle Time (how fast?) measures. All these measures rely on the batch size of the capability or stories being released, and the efficiency of the pipeline in generating and provisioning products. Automation of the build/test elements has a profound impact on all these measures. Consistency of staffing and team composition can also impact the team's ability to release their capabilities as needed. Generally faster release cycles on a predictable cadence are desirable to quickly deliver value to users and obtain feedback. There can be a tension between speed and quality tradeoffs. An over-emphasis on speed can be at the expense of product quality. There is often a 'sweet spot' tradeoff between speed and quality that delivers a best value solution based on project objectives. Quality needs to be monitored, in addition to speed, to ensure that these measures are appropriately balanced.  Additional statistical measures can be generated (e.g., mean, median, standard deviation, quartiles) to determine the aggregate performance, repeatability, and consistency of product release timelines.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Implementation Considerations	Applying Build/Test Automation to generate releases as early in the program as possible is recommended. Successfully generating releases as early in the release cycle should be a team priority. Integrity of the product baseline can be ensured by enforcing quality criteria for baseline merges to proceed successfully through the build/test automation pipeline.

Additional Specification Information	
Information Category	Process Performance
Measurable Concept	Process Efficiency – Speed
Relevant Entities	Releases, Effort
Attributes	Quantity, Labor Hours
Data Collection Procedure	Date/time is collected at the start and end of each iteration or release (iteration or deliverable, internal or external), typically obtained directly from automated tools. Each release must meet entry and exit criteria to be considered complete. Cycle time is calculated as the difference between release start time and release end time. Release frequency is calculated as the number of releases completed per unit time (e.g., day, week, month, year).
Data Analysis Procedure	Measures of the release process are analyzed at end of each release for performance within acceptable bounds, with corrective actions or improvements taken as necessary.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.10 Team Velocity (Team Measure)

Measure Introduction	
Description	Velocity is a measure of team performance and the amount of work that is completed in an iteration, typically a count of completed story points or equivalent. Velocity calculations can be used to estimate the amount of work that can be accomplished by the team in future iterations and when planned deliveries will be completed.
Relevant Terminology	Velocity — The average amount of work a team completes in an iteration or release. Used for planning and measuring team performance. Acceleration — Change in velocity across iterations.

Information Need and Measure Description	
Information Need	Is the team performing as expected? Does the team consistently meet the anticipated velocity? How much work can be accomplished by the team in a future iteration?
Base Measure 1	Story Points Completed (integer scale)
Base Measure 2	Iterations Completed (integer scale)
Derived Measure 1	Average Velocity = Story Points Completed / Iterations Completed
Derived Measure 2	Team Acceleration = (Current iteration Velocity – Reference Comparison iteration Velocity) / Reference Comparison iteration Velocity Note: the Reference Comparison iteration Velocity may be calculated as the Average Velocity across all teams, or by setting a goal for all teams to meet.
Derived Measure 3	Average Acceleration = Sum (Team Acceleration 1 …Team Acceleration N) / N

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Indicator Specification

Indicator Description and Sample

In Figure 21, Story Points Completed is graphed for each iteration [dark blue bars]. Average Velocity is then graphed as of each iteration [red line] based on last 4 iterations (4-iteration rolling average).

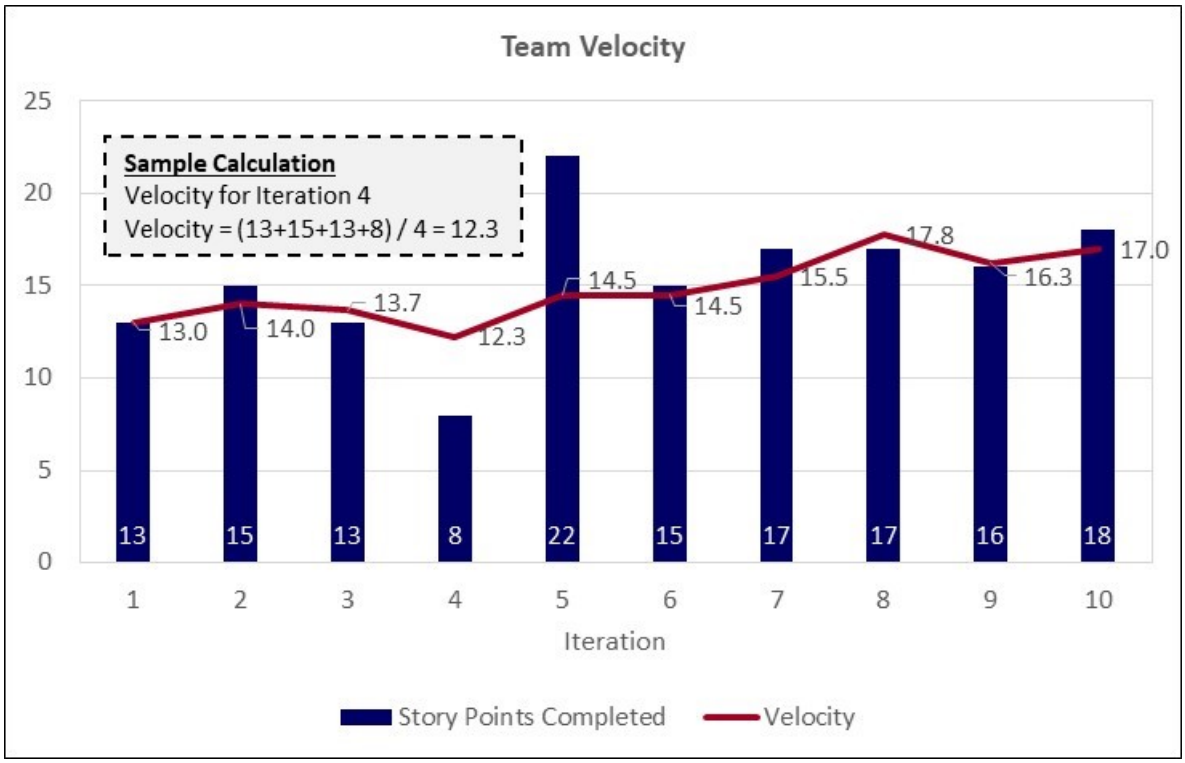


Figure 21: Team Velocity

Iteration 4 had a significant drop in velocity, followed by a large increase in iteration 5. This was due to several stories in iteration 4 that had defects. These defects were resolved in iteration 5, along with the completion of the iteration 5 assigned stories. Velocity improved and became more consistent after iteration 5, as the team became more experienced. This team established a consistent velocity after iteration 6.

Changes in velocity across iterations can be analyzed in more detail using acceleration measures. For instance, in Table 7, Teams 1, 2 and 5 show significant positive acceleration, which is typical for early iterations. Team 3 shows a dramatic drop, which should be analyzed to determine if there is a problem. Team 4 shows no variation, which may suggest a reporting anomaly.

Table 7: Sample Acceleration

	Iteration 1 Points	Iteration 2 Points	Acceleration
Team 1	10	12	20%
Team 2	8	9	13%
Team 3	14	8	-43%
Team 4	12	12	0%
Team 5	8	11	38%
Overall			5.6%

Sample Calculation
Team 1 Acceleration = (12-10) / 10 = 0.2
(20% positive acceleration)

Analysis Model

Do we have a consistent velocity? Why is the velocity changing over time? Based on past performance, is the average team velocity adequate to complete defined features allocated to this team? Variations may occur due to vacations, sick leave, changes to team size/composition, or implementation difficulties.

Decision Criteria

Velocity of +/- 10% should result in analysis at iteration review.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Additional Information	
Additional Analysis Guidance	Use this with the Committed Backlog and story point-to-feature ratio to ensure project will release identified features as scheduled (e.g., will velocity for remaining iterations be sufficient to complete committed features)? Will current average velocity be adequate to complete committed features by end of project? This assumes an ability to estimate average number of story points per feature (and then capability), based on performance. This measure can be used with Reference Comparison iteration Velocity for normalization. Acceleration can be tracked over time to develop predictive trends in performance. For example, performance tends to increase slowly in the first few iterations, then increase sharply, then plateau. Knowledge of long-term acceleration trends can enhance planning accuracy. Comparing individual team acceleration trends can highlight teams that have problems or that should serve as exemplars. Tracking program level acceleration trends is useful for bidding future work.
Implementation Considerations	In general, velocity is specific to a team and cannot be aggregated across teams to the project level. If velocity is normalized it can be used at the product or enterprise level. Usually, velocity should become more accurate and reliable over time as the team becomes more experienced, processes are established, data is regularly produced and reviewed, and the team gets better at estimating. Since story points may vary across teams, variations in velocity can be compared in percentage terms (positive or negative acceleration relative to prior reference iterations). This gives the program a way of determining which teams are struggling without having to normalize velocities.

Additional Specification Information	
Information Category	Process Performance
Measurable Concept	Process Efficiency - Speed
Relevant Entities	Features
Attributes	Stories, Story Points (estimated size)
Data Collection Procedure	Data is collected at the end of each iteration by the team lead from the team tracking tool. Story Points must be tested and satisfy the completion criteria, with no open defects to be counted as completed.
Data Analysis Procedure	Data is analyzed at the end of each iteration by the team during the iteration review and considered during the planning session for the follow-on iteration.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

8.11 Product Value (Team, Product, or Enterprise Measure)

Measure Introduction	
Description	The Practical Software and Systems Measurement (PSM) Product Value Measure (PVM) is a scalable, and flexible approach to measuring product value for a set of stakeholders. The PSM PVM includes product value measurement for three stakeholder groups; User, Acquirer, and Supplier. Product value can have different meanings for different stakeholders and depend on multiple product, or project, attributes. The results can be compared across products, projects, or initiatives if the components used in the measure are kept consistent. This measurement approach can work for software, hardware, systems, or projects. The defined objective and the related stakeholders will determine and prioritize the product attributes to be evaluated for a particular product or project. A common set of product attributes and common stakeholder groups are included in the baseline PSM PVM. Related sets of attributes are grouped into attribute categories. Additional attributes and stakeholders can be added to the measure as needed to satisfy the objectives of the measurement. Other attributes may include cost of producing value and product value impacts due to release delays. It is recommended to perform the PSM PVM on capabilities, software releases, or products during the planning stage to identify which options provide the most value to the stakeholders. Then repeat the measure using the same attributes at release to the customer and periodically during initial operation.
Relevant Terminology	**Cost of Value** is the cost associated with providing product value and satisfying the stakeholder attributes. This would be an estimate of lifetime cost divided by the estimated life span of the product. **Impact of Delay** is the estimated impact on Product Value based on delaying development and release of a product, system, or capability. This impact is based on estimated cost, or profit, related to the stakeholder group. Example: the impact of delay to a supplier may affect their income or profit. **Product Attributes** are characteristics of the product, system, or capability that are important to a set of stakeholders. Product Attributes may be important to more than one stakeholder. They describe the right product for the stakeholders and provide criteria for the evaluation. Stakeholders select 1-n attributes from each Product Attribute Category based on the measurement objectives. **Product Attribute Assessment** is a value calculated by related data, or assigned by one or more SMEs, to indicate satisfaction of the product to the stakeholder's attribute criteria. **Product Attribute Categories** are groups, or collections, of related key Product Attributes. The following six Product Attribute Categories are included in the specification of PSM Product Value Measure and cover the 31 product attributes included in this specification. **Usability and Operability:** Ability of a product, system, or capability, to be easy to use and operate and effectively utilize personnel resources such as manpower and skills. This includes learnability, operability, user error protection, user interface, and accessibility. **Performance:** The degree to which a system or component meets or exceeds technical requirements or delivery of capability that meet mission objectives with efficient system response and resource utilization measured or estimated under specified testing and / or operational environmental conditions. This includes time behavior, resource utilization, and capacity. **Functionality:** Ability of a product, system, or capability, to provide or facilitate all the specified tasks and user objectives with the correct results and the needed degree of precision; and meet mission capability needs. This includes completeness, correctness, and appropriateness. **Dependability:** Ability of a product, system, or capability, to consistently perform its intended functions over time, recover from any failure condition, be available and operable when needed. This includes availability, reliability, recoverability, maintainability, and maintenance support.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	Security: Ability of a product, system, or capability, to resist cyber and/or physical interruption, intrusion, spoofing, or degradation of its expected operation and functionality. **Business Value:** Ability of a product, system, or capability, to satisfy customer initial and total cost targets; supplier contract performance, including product delivery when promised; and supplier financial expectations throughout its lifecycle. **Product Attribute Weight** is a value between 0 and 100 indicating importance of the attribute to the stakeholders. The sum of all attribute weights for a product evaluated by a SME is 100 indicating 100% of possible value. **Product Value** is an assessment of the degree to which the delivered product, capability, or service satisfies, or will satisfy the needs of its stakeholders including but not limited to mission improvements, efficiencies, risk reduction, and cost. **Stakeholder** is a group, or individual, that has vested interest in the Product Value, i.e. User, Acquirer, Supplier. An organization may have multiple interests in a product and therefore they may belong to multiple stakeholder groups. For example: a company may be a supplier, acquirer, and user of a product. In this measurement they are considered different stakeholder perspectives. **User**: perspective of the end user or operator **Acquirer**: perspective of the purchasing organization, or buyer of the product **Supplier**: perspective of the company or organization that develops and sells the product to the Acquirer **Subject Matter Expert** (SME) is a person familiar with the product, capability, or system and is considered an expert by their management or industry.

Information Need and Measure Description	
Information Need	What value is being provided by the product, capability, or system? What is the Product Value to the stakeholders? Is the user satisfied with the delivered products? Do the products provide the desired functionality when needed?
Base Measures	
Attribute Category Weight	**Attribute Category Weight** (Wc) is set for each attribute category being used in the evaluation to represent the portion of the available value assigned to that category of attributes. If Attribute Category Weight is used, it is evenly distributed to all the attributes of the category. If the Attribute Category Weight (Wc) is not used, each attribute will be assigned its own Attribute Weight.
Attribute Weight	**Attribute Weight (Wa)** represents the portion of available value that is assigned to the specific attribute. The total weight of all attributes for an item being evaluated is 100 representing 100% of the value. Weights may be set to zero for attributes that are not contributing to the Product Value assessment. If Attribute Category Weight (Wc) is used, then the Attribute Weight is calculated as the Attribute Category Weight divided by the number of attributes in that category. A list of common attributes is provided later in this specification.
Product Attribute Raw Score	**Attribute Raw Score** (RSa) is set for each attribute being evaluated for an item and represent the satisfaction of the item with the attribute criteria. The range for the raw score should be 0-100 for each attribute with 0 representing none of the criteria of the attribute are satisfied and 100 representing all the criteria of the attribute are satisfied. The raw score for an attribute should be calculated based on quantitative data when applicable. A list of common attributes is provided later in this specification.
# SMEs	**# SMEs** is the number of SMEs involved evaluating the attributes

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Derived Measures	
Stakeholder Attribute Value	**Stakeholder Attribute Value (SAVs)** is the value achieved by an Item for a specific stakeholder group based on the selected attributes. This is calculated by adding the total attribute points achieved for all attributes related to the stakeholder group and dividing by the sum of the total maximum attribute score available for all attributes related to the stakeholder group. For an Item this represents the % of attribute satisfaction and is calculated for each Stakeholder Group (User, Acquirer, Supplier). Attribute points achieved is calculated as the attribute weight times the attribute raw score. Available score for an attribute is calculated by the attribute weight times 100. **SAVs =** Score achieved / Score available where Score Achieved = $\sum_{Attribute}(\sum_{SME}(W_{Attribute} * RS_{Attribute})) = \sum_{Attribute}(W_{Attribute} * \sum_{SME}(RS_{Attribute}))$ and Score available = $\sum_{Attribute}(\sum_{SME}(W_{Attribute} * \text{Max Score})) = \#\text{ SMEs} * 100 * 100 = \#\text{ SMEs} * 10000$ for all attributes selected by a Stakeholder group. This is calculated for each Stakeholder group. Where: 1. $\sum_{SME}(RS_{Attribute})_{SME}$ is the sum of raw score across all SMEs and is calculated for each attribute 2. $W_{Attribute} * \sum_{SME}(RS_{Attribute})_{SME})$ is the sum from step 1 times the attribute weight and is calculated for each attribute 3. $\sum_{\mathbf{Attribute}} (W_{Attribute} * \sum_{SME}(RS_{Attribute})_{SME})_{attribute}$ is the sum of the weighted scores across all attributes 4. $\sum_{Attribute}(\sum_{SME}(W_{Attribute} * \text{Max Score})) = \sum_{Attribute} W_{Attribute} * 100 * \#\text{ SMEs}$ $= 100 * 100 * \#\text{ SMEs} = (\#\text{SMEs} * 10000)$ since Max Score = 100 and sum of all attribute weights = 100.
Stakeholder Category Value	**Stakeholder Category Value (SCVs)** is the value achieved by an Item for a specific stakeholder group for each attribute category. This is calculated by adding the total attribute points achieved for all attributes in a category and dividing by the sum of the total attribute score available for all attributes in the category. For an Item this represents the % of attribute category satisfaction and is calculated for each attribute category and each Stakeholder Group (User, Acquirer, Supplier). Attribute points achieved is calculated as the attribute weight times the attribute raw score. Available score for an attribute is calculated by the attribute weight times 100. **SCVs** = Score achieved / Score available where Score Achieved = $\sum_{Attribute}(W_{Attribute} * \sum_{SME}(RS_{Attribute})_{SME})$ and Score available = $\#\text{ SMEs} * (100 * 100) = (\#\text{SMEs} * 10000)$ for all attributes selected by a Stakeholder group. This is calculated for each Stakeholder group. Where: 1. $W_{Attribut}$= Attribute Category Weight (Wc) / # attributes in the category and is calculated for each attribute 2. $\sum_{SME}(RS_{Attribute})_{SME}$ is the sum of raw score across all SMEs and is calculated for each attribute 3. $W_{Attribute} * \sum_{SME}(RS_{Attribute})_{SME})$ is the sum of step 1 times the attribute weight and is calculated for each attribute 4. $\sum_{\mathbf{Attribute}} (W_{Attribute} * \sum_{SME}(RS_{Attribute})_{SME})_{attribute}$ is the sum of the weighted scores across all attributes.

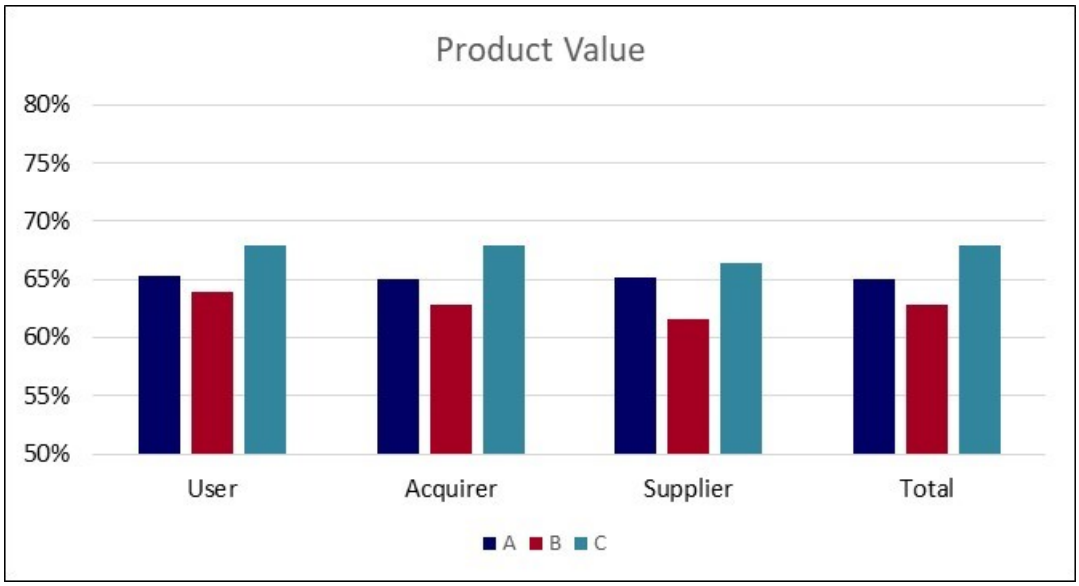
Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Product Value	**Product Value (PV)** represents the value achieved by an Item for all attributes. This is calculated by adding the total attribute points achieved for all attributes and dividing by the sum of the total attribute score available for all attributes. For an Item this represents the % of attribute satisfaction. Attribute points achieved is calculated as the attribute weight times the attribute raw score. Available score for an attribute is calculated by the attribute weight times 100. PV = Score achieved / Score available where Score achieved $=\sum_{Attribute}(\sum_{SME}(W_{Attribute}*RS_{Attribute})_{SME})$ and Score available = # SMES * (100 * 100) = (#SMEs * 10000) Where: 1. $\sum_{SME}(RS_{Attribute})_{SME}$ is the sum of raw score for all SMEs and is calculated for each attribute. 2. $W_{Attribute}*\sum_{SME}(RS_{Attribute})_{SME})$ is the sum of raw scores from step 1 times the attribute weight and is calculated for each attribute. 3. $\sum_{Attribute}(W_{Attribute}*\sum_{SME}(RS_{Attribute})_{SME}))$ is the sum of weighted scores across all attributes.

Indicator Specification	
Indicator Description and Sample	**Stakeholder Attribute Value (SAV)** is calculated for each stakeholder group (user, acquirer, and supplier). The results can be represented in a tabular format or using a column graph as depicted below. The first table and graph represent an example SAV for each stakeholder group. **Table 8: Sample Stakeholder Attribute Values**

A	B	C	
65.37%	63.88%	67.89%	User
64.97%	62.89%	67.93%	Acquirer
65.11%	61.58%	66.38%	Supplier
64.97%	62.89%	67.93%	Total

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	 **Figure 22: Stakeholder Attribute Value** Assessments of Stakeholder Attribute Value and Product Value should be made during planning and in preparation for release, and periodically after deployment. These should be compared to each other to understand any trends in product value.
Analysis Model	A specified level of Product Value may be set based on historical assessments of similar capabilities or systems. Such a specified level should be agreed to by the stakeholders. When comparing Product Value across capabilities, products, or systems for prioritizing future work the analysis should identify the item with the highest attribute value. The Product Value Measure should be performed at a capability, or Epic level and above for software products. Analysis should include sensitivity analysis to assess the robustness of the results of the measure.
Decision Criteria	Decision criteria and method should be determined by the stakeholder groups before the assessment is made. Decision criteria should be measurable and support decisions on different stakeholder courses of action. For example, the decision criteria may be to select the item with the highest total attribute value. It may also be to select the item with the highest attribute value related to a specific stakeholder or category.
Indicator Interpretation	Identify the percent of attribute satisfaction.

Additional Information	
Additional Analysis Guidance	Use actual data to assess satisfaction of the attributes whenever possible. When assessing the same product for trending purposes make sure the attribute weights are kept consistent from one periodic assessment to the next. This forces the changes in value to be driven by changes in how the attributes are satisfied. When assessing different products for comparison, make sure the attribute weights are kept consistent across the products being assessed. This forces the changes in value to be driven by differences in how the attributes are satisfied.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Implementation Considerations	All programs/projects should have a strategy for project assessment and control. The product value assessment process should be enabled during planning. Initial assessments are conducted to baseline the assessment. After fielding, periodic assessment should be conducted to identify trends and issues. Program/projects may choose one or two attributes from each of the six attribute categories for simplification. Program/projects may use Attribute Category Weights as defined above for simplification. PSM provides a worksheet to assist implementation of the PSM PVM.

Additional Specification Information	
Information Category	Product Quality and Customer Satisfaction.
Measurable Concept	Product Value is measured as the satisfaction of specified product attributes.
Relevant Entities	Capabilities, software products, or systems.
Attributes	Attributes of Usability and Operability weight and score. Attributes of Functionality weight and score. Attributes of Performance weight and score. Attributes of Dependability weight and score. Attributes of Security weight and score. Attributes of Business Value weight and score.
Data Collection and Analysis Procedure	1. Prepare for the PV measurement a. Identify the item being evaluated. b. Identify the Stakeholders, their applicable Stakeholder groups, and the SMEs performing the measure. c. Establish the objective or purpose of the evaluation and how the results will be used. d. Select the attributes to be evaluated for the measure. e. Establish the weights for each attribute selected. Weight of each attribute will be set by stakeholder agreement. 2. Distribute information to the SMEs performing the evaluation. 3. SMEs will set the raw score for each attribute for each item being evaluated representing how well the item satisfies the attribute criteria. Assessed satisfaction of each attribute will be determined by estimated or actual data, or if data is not available by expert opinion. General criteria for satisfying each attribute is provided in this PSM PV Measurement Specification. 4. Collect the raw scores from the SMEs performing the attribute evaluation. 5. Calculate the Stakeholder Attribute Value and Total Product Value and prepare the results. The calculations are based on the formulas described in the Derived Data section of this Product Value Measurement Specification. 6. Product Value results are provided to the stakeholders.
Product Value Attributes, Criteria, and Categories	
Usability and Operability Attributes	1. Is the system, product, or capability easy to use and operate? – Degree of ease of operator use. Is additional training needed? Does it reduce amount of work and time required to do the mission or objective? Is it intuitive to use? Does it adapt to user mistakes? Does it include user guidance? Does it reduce the number of user steps to complete the task? Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; and Acquirer.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	0 = Additional training is needed. Increases the amount of work and time required to do the mission or objective. Not intuitive to use. Does not adapt to user mistakes. Does not include user guidance. Increases the number of steps to complete the task. 50 = No training is needed. No increase in amount of work and time to do the mission or objective. Does not improve or diminish intuitive use. Adapts to some user mistakes. Manual user guidance is included. Does not increase or decrease steps to complete task. 100 = Reduces need for training. Reduces the amount of time to complete the mission or objectives. Improves intuitive use. Prevents or adapts to user mistakes. No guidance is needed to use. Decreases steps to complete tasks. 2. Are manpower, skills, and resources available to execute and maintain the system, product, or capability? - Degree the system, or capability, can be operated and maintained with the planned operational and support resources. Does it take more or less effort to use? Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended stakeholders: User; and Acquirer. 0 = There is an increase in manpower, skills, and other resources to operate and maintain the system, or capability. 50 = There is no increase or decrease in operational or maintenance resources and or skills. 100 = The system, or capability can be operated and maintained with fewer resources than expected. 3. Does the user have appropriate access to the system, product, or capability? - Degree of access the operator or user needs in order to operate and use the system. Set the Value 0 – 100. Recommended Stakeholders: User. 0 = Additional user access is needed, or additional admin access is needed, or additional maintenance access is needed. 50 = There is no change in access needed for user, admin, or maintenance. 100 = Reduced access for user, admin, and/or maintenance.
Functionality Attributes	1. Does the system, product, or capability work as intended or required? – Degree the system, or capability, operates as expected, or required, in its intended environment. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; Acquirer; and Supplier. 0 = There is limited to no execution, no requirements are satisfied, no new or improved functions, and reduced operational performance. 50 = Satisfies some requirements and some operations are as intended. Some mission capabilities are provided. 100 = Provides new or improved mission capabilities, functions, or features and performance which meets or exceeds those requested or required. 2. Does the system, product, or capability satisfy or improve mission needs? - Degree the system, or capability, satisfies the users mission, objective, or purpose. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; and Acquirer. 0 = The system, or capability, place mission objective at unacceptable risk. The system, or capability has reduced mission capabilities and performance. 50 = Some mission objectives are satisfied but no improvement to mission capability. 100 = Improved performance and interoperability with improved mission capability and reduced risk.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	3. Does the system, product, or capability, meet all contractual requirements or Capability Needs Statement? – Degree the system, product, or capability meets the contractual requirements imposed by the acquirer. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: Acquirer; and Supplier. 0 = The system, product, or capability does not meet most contractual requirements and was not accepted by the acquirer. 50 = Some contractual requirements are met but the system was accepted with workarounds by the acquirer. 100 = The system, product, or capability meets or exceeds all contractual requirements. 4. Does the system, product, or capability, align with the product roadmap or known future needs? – Degree by which the system, product, or capability satisfies or is consistent with the acquirer's product roadmap. Set the value 0-100. Recommended Stakeholders: User; Acquirer; and Supplier. 0 = The system, product, or capability is not consistent with acquirer's roadmap, long and short-term goals. 50 = The system, product, or capability is consistent with short-term goals, but only some long-term roadmap objectives. 100 = The system, product, or capability is consistent with all acquirer's roadmap objectives. 5. Are there operational or sustainment issues with the system, product, or capability? – Degree by which the system, product, or capability is free from any known operational or sustainment issues. Set the value 0-100. Recommended Stakeholders: User; and Acquirer. 0= The system, product, or capability increases operational, maintenance, or sustainment issues. 50 = The system, product, or capability does not increase or decrease operational, maintenance, or sustainment issues 100 = The system, product, or capability reduces the known operational, maintenance, and sustainment issues. 6. Is the release cadence to push new capability to the field reasonable and acceptable? – Periodic releases of new capability will meet user needs. Set the value 0-100. Recommended Stakeholders: User, Acquirer and Supplier 0= The release cadence does not meet user or acquirer needs. 50 = The release cadence meets either user or acquirer needs but not both. 100 = The release cadence meets both user and acquirer needs.
Performance Attributes	1. Does the system, product, or capability, perform to expected system measures of performance (MOP) and effectiveness (MOE) within expected, or contractual, system resource limitations? – Degree by which the system, product, or capability performs its intended functions and operations efficiently within target resource constraints. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; Acquirer; and Supplier. 0 = The system, product, or capability decreases performance of some capabilities or increases system resource needs. 50 = The system, product, or capability does not improve or degrade performance of any capabilities or system resource needs. 100 = The system, product, or capability improves capability performance within system resource limitations or reduces resource needs.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

2. Does the system behave gracefully when approaching resource limits such as large number of users or transactions or increased demand? – Degree by which the system, product, or capability can continue to perform its intended functions as user demands or number of transactions increase. Set the value 0-100.
Recommended Stakeholders: User; and Acquirer.

   0= The system, product, or capability has degraded functionality and performance as the user demand increases or the number of transactions increase.

   50 = There is some degradation in performance or functionality as user demand increases or the number of transactions increase.

   100 = The system, product, or capability continues to perform with no degradation as user demand or number of transactions increase.

3. Does the system, product, or capability provide the results within expected, or needed response time? – Degree by which the system, product, or capability provides the results, actions, or responses within contractual or expected response time. Set the value 0-100. Recommended as part of a minimal set of attributes.
Recommended Stakeholders: User.

   0 = The system, product, or capability does not provide any results within expected response time.

   50 = The system, product, or capability supports degraded operations within expected response times.

   100 = The system, product, or capability provides all results within expected and needed response time.

4. Does the system, product, or capability meet or exceed the most important specified mission technical performance objectives, thresholds, or properties in an operational environment? – Degree by which the system, product, or capability can meet its specified mission technical objectives, thresholds, or properties while in its expected operational environment. Set the value 0-100. Recommended as part of a minimal set of attributes.
Recommended Stakeholders: User; and Acquirer.

   0 = The system does not meet any of its key specified mission objectives while operating in its expected environment.

   50 = The system meets about half of the key specified mission objectives in its expected environment but about half are not met.

   100 = The system meets or exceeds all the most important specified mission objectives while operating in its expected environment.

5. Does the system provide enough margin for future growth in performance required to accommodate anticipated future mission needs? – Degree by which the system, product, or capability allows for future growth in performance. Set the value 0-100.
Recommended Stakeholders: Acquirer; and Supplier.

   0 = The system has no performance margin for contractual, or expected, growth in performance or capacity for known future needs.

   50 = The system has about half contractual, or expected, performance margin allowance for growth for known future needs.

   100 = The system meets all contractual or requested performance margin allowance for future growth for known future needs.

6. Is the downtime to perform upgrades or maintenance reasonable and acceptable? – Degree by which the downtime to perform upgrades and maintenance affect performance. Set the value 0-100.
Recommended Stakeholders: User and acquirer.

   0 = The system cannot perform during upgrade or maintenance events or down-time is unacceptable.

   50 = The system has minimal operation but acceptable down-time during upgrade or maintenance events.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	100 = The system is fully operational with no down-time during upgrade or maintenance events.
Dependability Attributes	1. Is the system, product, or capability reliable and available when needed? – Degree of impact of failures, shutdowns, system locking up, or waiting on system to the user, mission, or objective. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; Acquirer; and Supplier. 0 = System failures, shutdowns, locking, and operational delays prevent efficient or effective operation. 50= System failures, shutdowns, locking and delays happened occasionally and precludes execution of some missions with existing resources or requires excessive resources to meet mission objectives. 100 = There are no failures, shutdowns, locking, delays or degradation in operation exhibited by the system. 2. Did you get the system, product, or capability when you needed it? - Ability to rapidly deliver, update, and/or fix system, or capability to meet operational needs. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; Acquirer; and Supplier. 0 = Delivery, update, and fix capability does not meet operational needs. 50 = About half the delivery, update, or fix capabilities provided meet operational needs. 100 = All expected mission capabilities, functions, features, and performance were delivered, updated, or fixed to meet operational needs. 3. Does, or will, the system, product, or capability life expectancy meet contractual or customer needs? – Degree the system, or capability life expectancy meets planned mission or user needs. This may also relate to product roadmap. Set the value 0-100. Recommended Stakeholders: Acquirer; and Supplier. 0 = Product life expectations will not meet mission or user needs and cannot be extended. 50 = Product life expectations meet current contractual or customer needs but there is no cost-effective means to extend its life. 100 = Product life expectations exceed mission and user needs and can be extended as needed. 4. How easy does the system, product, or capability recover operation from failure mode? – Ability of the system, product, or capability to recover normal or degraded operation as the result of a failure. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; and Acquirer. 0 = The system, product, or capability cannot recover any operations as a result of a failure. 50 = The system, product, or capability can recover to degraded operations. 100 = The system, product, or capability recovers full operation automatically after a failure. 5. How easy can the system, product, or capability be developed? – The degree of difficulty of development of the system, product, or capability due to technical issues or technical maturity. Set the value 0-100. Recommended Stakeholders: Acquirer; and Supplier. 0 = Technical maturity or issues make the system, product, or capability extremely difficult to develop. 50 = Technical maturity and lack of technical issues make the development of the system, product, or capability moderately difficult to develop. 100 = Technical maturity and lack of technical issues make the system, product, or capability easy to develop. 6. Does the system, product, or capability provide enough information, detail, or resources to be maintained during operation? – Degree of information, detail, or resources provided to support maintenance during operations. Set the value 0-100.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	Recommended Stakeholders: Acquirer. 0 = No guidance, information, detail, or resources are provided to support maintenance. 50 = Some guidance, information, detail, and resources are provided so maintenance can be done but not easily. 100 = Guidance, information, detail, and resources are provided for easy maintenance. 7. Is the corresponding end-of-life for hardware and other components of the system reasonable and acceptable? – Degree by which all the components of the system have appropriate life expectancies. Set the value 0-100. Recommended Stakeholders: Acquirer and supplier. 0 = None of the external components have acceptable or reasonable end-of-life. 50 = About half of the external components have acceptable or reasonable end-of-life. 100 = All external components have acceptable end-of-life.
Security Attributes	1. Is the system, product, or capability secure to use? Context: Degree that the system, or capability protects the user and data from harm. Recommended Stakeholders: User, Acquirer, Supplier. 0 = Security controls are ineffective, or not provided and prevents, degrades, and/or places operation at unacceptable risk. 50 = Some security controls are included but operations is still at risk but has been accepted by the user. 100 = Improved security controls reduce mission risk. 2. Does the system, product, or capability resist cyber and/or physical interruption, intrusion, spoofing, or degradation of its intended functionality and operation? - Degree by which the system, product, or capability can prevent or resist any interruptions in normal operations due to external influences. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; Acquirer; and Supplier. 0 = The system, product, or capability has no resistance to any physical and cyber interruption, intrusion, spoofing or degradation of its intended functionality. 50 = The system, product, or capability resists some physical and cyber interruption, intrusion, spoofing, with some degradation of performance. 100 = The system, product, or capability resists all known physical and cyber interruptions, intrusions, spoofing with no degradation of its functionality or performance. 3. Is the system, product, or capability, vulnerable to security attacks? - Degree of which the system, product, capability resists, or prevents security attacks. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: User; Acquirer; and Supplier. 0 = The system, product, or capability has no resistance to security attacks, and they may cause shutdown. 50 = The system, product, or capability has some resistance to security attacks, but operation is still at risk. 100 = The system, product, or capability has built in resistance and preventive measures to security attacks with no degradation in operation. 4. Is the approach for recurring accreditation reasonable and acceptable? - Does the approach for renewing security accreditation meet needs of the user and acquirer? Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: Acquirer; and Supplier.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	0 = There is no approach or plan for recurring accreditation. 50 = There is an approach for recurring accreditation, but it is not acceptable. 100 = The approach and plan exist and are acceptable to both user and acquirer.
Business Value Attributes	1. Will the system, product, or capability, improve mission needs while meeting or exceeding project budget constraints? - Degree by which the system, product, or capability will improve the mission capability and yet stay within budget constraints. Set the value 0-100. Recommended as part of a minimal set of attributes. Recommended Stakeholders: Acquirer; and Supplier. 0 = The system, product, or capability does not improve mission needs and is not expected to perform to budget constraints. 50 = The system, product, or capability may improve mission needs but not perform to budget or may perform to budget but not improve mission needs. 100 = The system, product, or capability will improve mission needs and is expected to perform well within budget constraints. 2. Does the system, product, or capability, add to supplier portfolio and market share? - Degree of business impact and product portfolio. Is this a new line of business or product line? Set the value 0-100. Recommended Stakeholders: Supplier. 0 = There is no market or portfolio advantage for system, product, or capability. 50 = There is continuing market for the product and/or is an important part of the supplier's portfolio but no future investment. 100 = There are market demands and portfolio advantages for the system, or capability, and is, or will be, available to meet market demands and the supplier will continue to invest and improve. 3. Does the system, product, or capability have financial value for the supplier? - Degree of financial impact to the company (Cash flow, revenue, profit…) or the ability of the organization to support the project with their current budget and resources. Will this positively impact company financial standing? Set the value 0-100. Recommended Stakeholders: Supplier. 0 = Supplier will suffer severe negative financial impacts or loss of organization's ability to support the system 50 = The is no financial advantage to the supplier. 100 = Supplier would make financial gains with potential for future business or the organization can provide significant increases support with lower cost and resources 4. Is the system, product, or capability, cost effective to produce? - Degree of cost investment versus return for the company /organization efficiency/effectiveness? (Return on Investment) Set the value 0-100. Recommended Stakeholders: Acquirer; and Supplier. 0 = Investment will not be recovered in short term or long term. 50 = The cost of investment and return cancels each other. No loss or gain. 100 = Return on investment is better than expected. 5. Is there an impact to value due to delay in delivery? Separate value for supplier, acquirer, and user impacts. - Degree of impact to the value of the system, product, or capability if it is delayed compared to its potential lifetime value. the value 0-100. Recommended Stakeholders: User; Acquirer; and Supplier.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

	0 = Significant impact to the stakeholder's business model. I.E. loss of future orders/profit/revenue/market leadership, cancelled deployments, delayed retirement of other systems, loss of program funding… 50 = Some impacts to the stakeholders' business model I.E. delays and or reductions in orders/profit/revenue, reduced market share, delayed deployments, reduced program funding, delays in fielding follow on capabilities 100 = Minimal or no impact to stakeholder business models

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

9. Enterprise Measures

Measures collected at the team and product levels are typically driven, at least in part, by flow down of enterprise business objectives and information needs for management oversight and enterprise decision making. Team and product measures are used not only for insight and action at those respective levels, but also to meet the higher-level needs of the enterprise. This includes the need to provide governance over enterprise programs, and ensure effective execution of those programs. Measures are often summarized, aggregated, or transformed to provide insight to managers and executives with responsibility for the performance, competitiveness, risks, and growth of the business unit or enterprise overall, to address questions such as:

- How are our projects or products performing?
- Are we meeting commitments?
- What is the quality of products or services we deliver across the enterprise?
- Is productivity improving?
- How accurate are our estimates?
- What is the process efficiency for programs, businesses, or products?

At the enterprise level, executives may have additional concerns relative to the business overall beyond a roll-up of project level measures, related to areas such as business strategy, technological enablers, resources, and workforce capability.

This alignment of concerns and perspectives is illustrated in **Figure 23**. Team and product measures (such as those in Section 8) are used by teams or projects to manage their daily work and performance against their objectives, using measures obtained from native tools and processes. These measures are then collected at higher levels of the enterprise to provide managers with insight across projects for monitoring, decision making, and management action as appropriate. This often requires transformation of disparate project level measures to a common standard for reporting using indicators specifically designed for appropriate management insight, analysis and actions at enterprise levels, as depicted in the examples shown in Figure 23. These measures must be aligned and compatible at all levels of the enterprise.

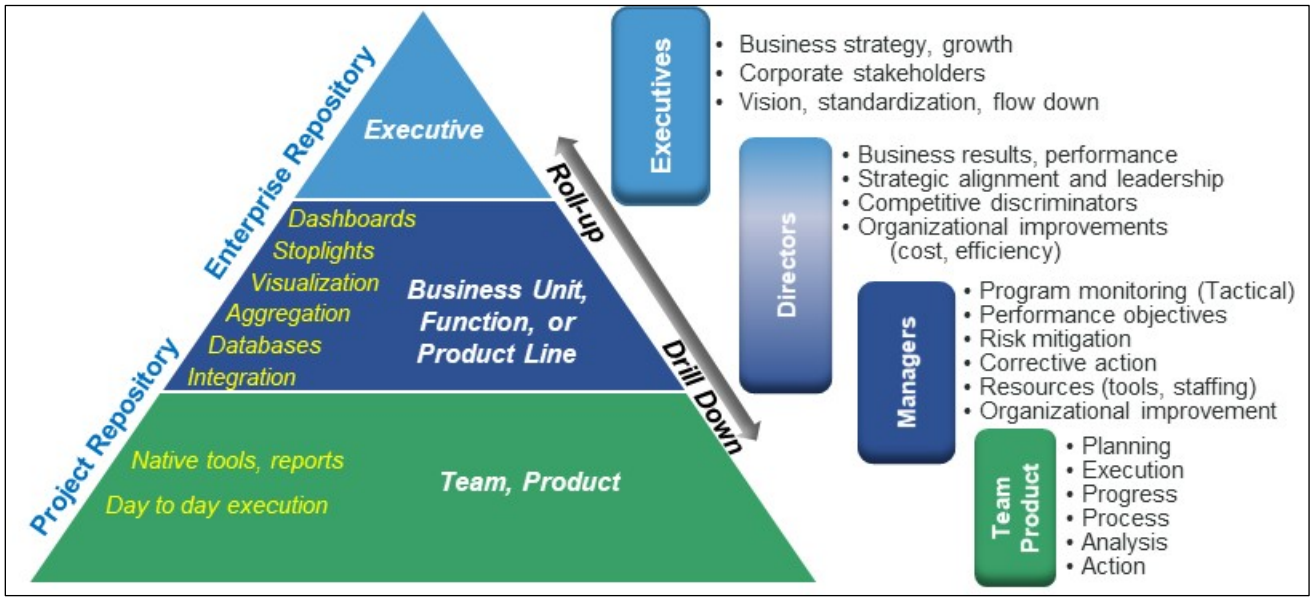


Figure 23: Alignment of Enterprise Measures

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

The ICM table in section 7 lists candidate enterprise information needs aligned with PSM information categories, measurable concepts, and potential measures. Section 8 provides examples, indicators, and measurement specifications for a prioritized set of measures selected from this list. Some of these measurement specifications already provide examples of enterprise measures derived from measurement data at the project or product level. This section provides additional representative examples, based on the measures described in section 8. The measures an enterprise chooses to implement and collect will be tailored based on alignment with its information needs and objectives, so they may differ from those described here. Additionally, the enterprise may identify additional measures relative to its business strategy beyond the set described here, such as measures of transition, adoption, workforce development, or market share.

This section focuses on example indicators and challenges appropriate to data collection, analysis and reporting of CID measures at the enterprise level. The information provided in the section 8 measurement specifications will generally not be repeated here, nor will separate specifications be generated. This is because much of the content defined there is applicable and relevant to enterprise measures, and indeed derived from it, including:

- Description and relevant terminology
- Definitions of base and derived measures – used to ensure the consistency and integrity of measures aggregated to the enterprise level
- Analysis model – how to obtain insight from the indicators provided
- Decision criteria – guidance for acting on indicator trends
- Additional analysis guidance
- Implementation considerations
- Data collection and analysis procedures – how the project measures are collected to provide valid data to the enterprise level

Refer to the applicable section 8 specifications when considering the implementation of the enterprise measures and indicators suggested. Some additional guidance specific to enterprise measures is provided here, such as information needs, indicators, analysis, trending, and potential management insight or actions.

But first we must consider some of the challenges in collecting, transforming, analyzing, reporting, and acting upon enterprise measures, especially the unique challenges of integrating data from CID projects.

9.1 CHALLENGES FOR CID ENTERPRISE MEASURES

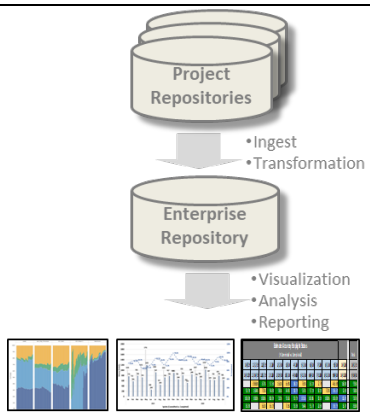
As depicted in **Figure 23**, the enterprise measurement framework depends on alignment of business needs with measures collected at the project level, supported by the ability to aggregate (roll-up) key measures and trends to functional and executive levels for summary insight and decision making, with an ability to drill down to lower levels for details when needed.

Aggregating CID measures to the enterprise level must potentially address issues and challenges with project-level measures, such as inconsistent schema definitions, integrating project data collected at differing reporting cadences, numeric conversion to data types that can be reasonably aggregated, and defining bounds and thresholds for ranges of expected performance. Some of these challenges are described in Table 9.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Some enterprises may have project measures collected directly into enterprise measurement repositories. Some may have separate project repositories that must be mined into aggregate enterprise reports or repositories. Additionally, some enterprise data is not project specific, such as workforce data or process improvement data. Vendor tool suites, plug-ins, or business intelligence (BI) analysis tools can help with collecting and aggregating measures from diverse repositories or platforms, or enterprises may develop their own tools, interfaces, and scripts to do so. Often a transformation layer is needed between the project and enterprise data to normalize data and allow cross-comparisons. But in all cases, automated collection of project measures as appropriate to the enterprise level is highly desirable.

Table 9: Challenges for Collecting and Reporting CID Measures at the Enterprise Level

Data Collection	An enterprise measurement infrastructure may be organized into architectural layers, such as: • Ingest: measures must be collected from project repositories at regular intervals, ideally automated. • Transformation: diverse project measures from different schemas or varying operational definitions may need to be transformed to common standards and definitions for enterprise level analysis. • Visualization: aggregation of native project level measures and indicators may not be effective representations for management insight across projects and assessment of overall enterprise performance. Alternative indicators may be needed, such as project summaries, dashboards, averages, or stoplights. 
Roles, Authority, and Accountability	Ownership and responsibilities must be defined with accountability for collection, analysis and reporting of source measures to the enterprise, including project staff, functional managers, enterprise leaders, and support groups. Processes and procedures should be established to ensure the data collected is accurate, timely, current, consistent, and validated. Plans should be established to ensure adequate resources are provided, such as sufficient budget, staff, tools, facilities, and IT infrastructure.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Varying Reporting Cadence

CID projects typically operate at a cadence of increments or releases which can vary across projects (e.g., days, weeks, months). Enterprise reporting is usually based on calendar months or fiscal periods (which may be 4-5 weeks in duration). This means the enterprise reporting may need to integrate project measures of varying frequency or amplitude, which can be problematic for monthly summaries or trends based on inconsistent data sets. This is illustrated in the following example of project data set reports. For simplicity, let's assume each project operates at a consistent output of 20 units per week, but reported upon completion at varying sprint duration cadences.

Project	Cadence (Sprints)	Fiscal Reporting Period 1				Fiscal Reporting Period 2				
		Wk 1	Wk 2	Wk 3	Wk 4	Wk 1	Wk 2	Wk 3	Wk 4	Wk 5
A	2 weeks	40		40		40		40		40
B	3 weeks		60			60			60	
C	4 weeks				80				80	
Org	Period	4 data pts, 220 units				6 data pts, 320 units				

This variation in reporting cadence can make aggregated reporting and trend analysis problematic at the enterprise level, even when project outcomes are stable and predictable. Reporting CID enterprise measures based on calendar months instead of fiscal periods can reduce but not eliminate reporting issues resulting from this inherent variability of project reporting cadences. Variation can also be moderated by monitoring rolling averages over time. Managers can also drill down to the project-level measures for analysis of the actual project performance at true cadence.

Inconsistent Project Measures

Sometimes enterprises will have standard operational definitions for CID measures consistently applied across projects. But some measures, such as story points, have team-specific units that cannot be aggregated directly to the project or enterprise level.

Other issues that prevent aggregation of project data include a lack of a consistent size measures that can be used to normalize the results, productivity variations across teams, and different scales for various base measures.

Enterprise Indicator Types

Enterprise insight can sometimes be obtained by statistical analysis of performance measures. To enable this, measurement indicators providing enterprise management insight may need to be standardized, transformed numerically to common units, or defined using alternative views. In the examples in section 9.2, three types of enterprise analysis indicators are illustrated:

• *Summary* analyses: a summary of measures displayed for a set of individual projects (not aggregated), collected in a convenient display for managers to easily scan measures across a set of programs and identify potential issues for drill down to the program level details. This indicator type may be most useful when the project-level cannot be directly combined.

• *Aggregate* analyses: project-level measures are integrated and aggregated across a set of projects (e.g., all programs combined into aggregate measures at the business unit level) to provide insight into performance trends of the enterprise overall.

• *Stoplight* analyses: executive summary of project performance depicted in color indicators (e.g., Red, Yellow, Green, Blue) relative to established performance ranges (thresholds). Stoplight ranges may be tailored to the needs of the decision makers. Stoplight charts can be viewed at the project level (by month), or at the executive level as counts of projects falling in each performance color range.

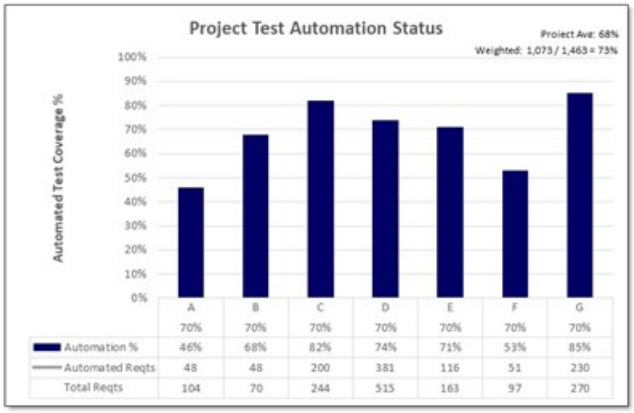
Selection of appropriate indicator types for enterprise measures and cross-project comparisons may depend on the nature of the underlying project source measures, characteristics, and the extent to which the measures are arithmetically consistent.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

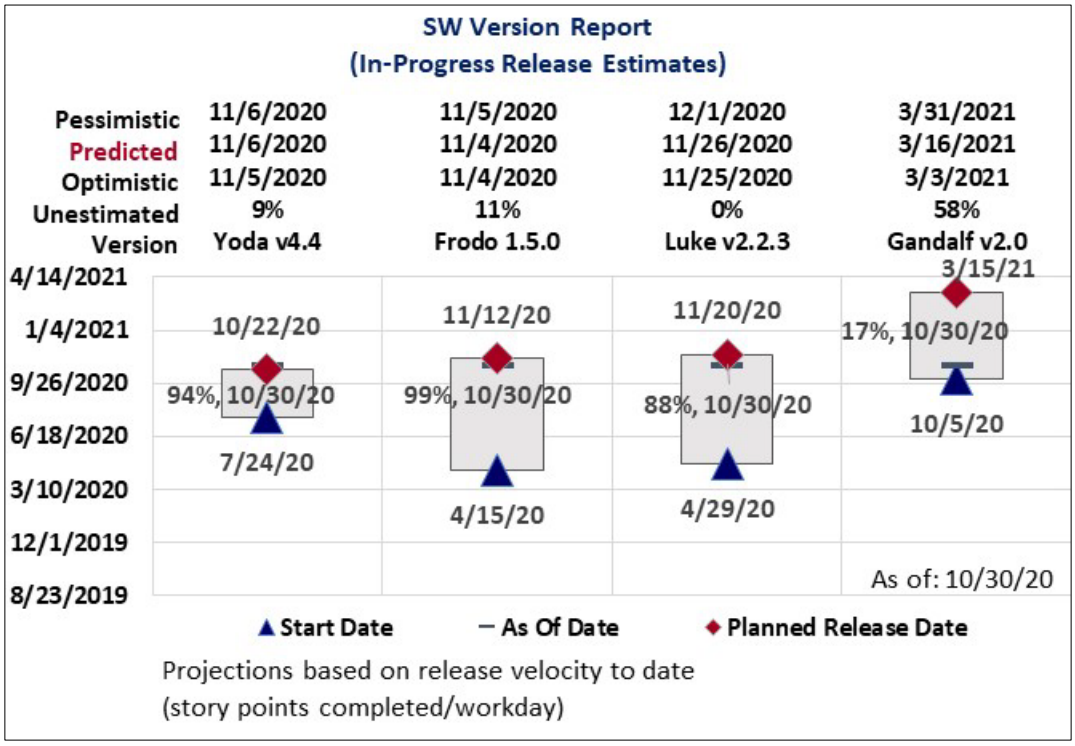
9.2 Example Enterprise Measurement Indicators

Selection of measures and indicators for enterprise reporting can depend on many factors, particularly the enterprise's business objectives, application domain, product types, and information needs. There is no one answer to these questions. The indicators and information needs suggested here are examples, based on excerpts of the PSM CID project measures in section 8, to help enterprises consider ideas that may be appropriate to their situation.

Table 10: Measurement Specification Mapping and Examples

Automated Test Coverage (8.1)	*What is the extent of automated testing conducted across the enterprise's projects? What benefits to enterprise performance (e.g., cycle time, quality, throughput) are enabled by effective automated testing?* At the enterprise level, indicators such as this example from Section 8.1 Figure 4 provide insight into the extent of automated test coverage across projects. In this example, a *Summary* indicator is used to show the average automated testing across projects (average of averages) is 68%. But the count of requirements (scale) varies across projects; the weighted average of the base measures is 73% (*Aggregate* indicator). In this example the enterprise has set a goal of ≥70% test automation, and this indicator depicts which projects are meeting that target and where a further deep dive may be needed.  A B C D E F G 70% 70% 70% 70% 70% 70% 70% Automation % 46% 68% 82% 74% 71% 53% 85% Automated Reqts 48 48 200 381 116 51 230 Total Reqts 104 70 244 515 163 97 270 The example summary indicator can be extended if needed for larger data sets by providing display filters or selectable project criteria, such as business areas, ranges of threshold values, counts of projects achieving or not achieving target thresholds over time, or other project characteristics. Executives may also want to monitor indicators of enterprise-wide adoption of automated testing across projects. There may be valid reasons why some projects may not be able to achieve the goal due to project constraints or cost-effectiveness analysis.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Burndown (Release) (8.2)	*How likely are we to deliver planned releases on time?* At the enterprise level, manager emphasis is on meeting deliverable commitments such as major releases. This example *Summary* indicator depicts upcoming release status across projects, based on a combination of Release Burndown (8.2) and Team Velocity (8.10) measures to predict a range of estimated release dates vs. plan based on % of story points completed and velocity achieved to date.  **Figure 24: Burndown - In Progress Release Report** In this example, project Yoda release v4.4 has already missed its planned release date of 10/22, with estimated release now on or around 11/6. The Frodo 1.5.0 project release is currently estimated to complete early. Luke v2.2.3 is on a pace to finish about a week behind, but at 88% complete and with all stories fully estimated, the team believes the recent addition of a test resource will enable them to still deliver on time in a few weeks. Gandalf v2.0 is a major product upgrade planned for release in 4½ months; it currently seems on plan, but 58% of the stories mapped to the release are not yet fully estimated, which may indicate significant risk. The example summary indicator can be extended if needed for larger data sets by providing display filters or selectable project criteria, such as business areas, ranges of target release dates (e.g., due within 30 days), projects estimated to significantly miss milestone dates, or other project characteristics. Tabular displays with release status by project (early, on time, late) could also be used. Often projects may defer functionality to subsequent releases in order to meet delivery schedules; release content can be monitored using other project-level drill down displays such as Committed vs. Completed. For any projects indicating release schedules are at risk, managers can drill down to project-level measures for additional details and implement corrective actions as needed. Other complementary measures in the core PSM guidance, such as milestone tracking and schedule margin, can be used to assess confidence and risk in meeting committed release dates.
Committed vs. Completed (8.3)	*How accurate are our estimates? Are commitments being met consistently (plan vs. actual)?* User stories and story points are typically project-specific measures by iteration with units that are not readily comparable or aggregated across projects. However, the % of stories or story points completed

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Estimate Accuracy

vs. committed (see 8.3, Figure 6 and below) is a relative measure of estimation accuracy that can be plotted over time to provide an indication of predictability. Similarly, an *Aggregate Indicator* (**Figure 25**) can be used to show the overall completion % at the enterprise level across projects. In this example, the primary indicator is the % Completed dashed line. The secondary indicators for Stories Committed and Stories Completed are shown only for reference to help interpret the context and magnitude of work completed in any given month, but the values cannot be compared at the enterprise level across projects due to variations in project iterations and cadence.

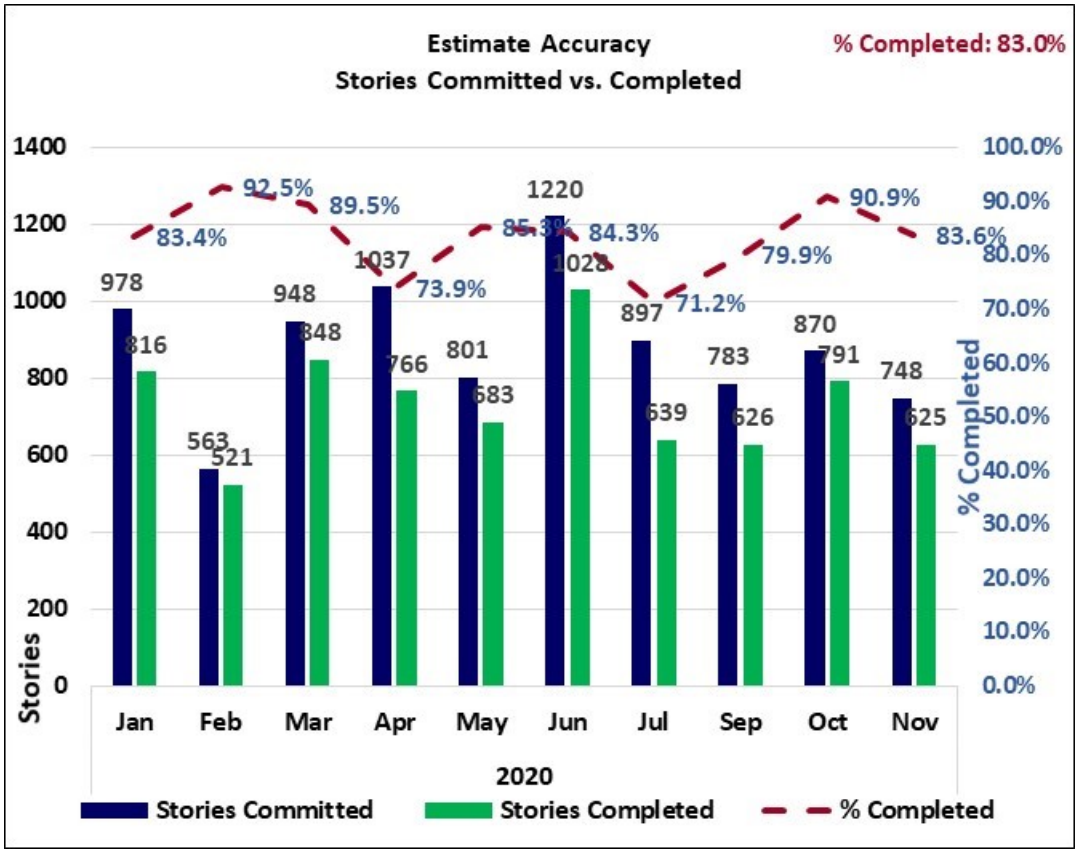


Figure 25: Aggregate Indicator - Stories Committed vs. Completed

A *Stoplight Indicator* can be used to summarize how well projects meet their planned commitments, as a percentage against criteria ranges (Red, Yellow, Green, Blue) such as those in Figure 26. Note that the color thresholds have two ranges, one for completed less than planned and one for completed more than planned. Either could have actions to improve predictability for future iterations. Note also that the enterprise wants projects to set aggressive stretch targets, say 80% completed vs. setting a low bar that is always met. Stories not completed roll over to the next iteration.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Iteration Estimate Accuracy - by Stoplight Threshold Criteria					
EA < 60%	60% < EA < 80%	80% < EA < 100%	100% < EA < 120%	120% < EA < 140%	EA > 140%

	Estimate Accuracy Stoplight Status (% Committed vs. Completed)												Total
Project	Oct-19	Nov-19	Dec-19	Jan-20	Feb-20	Mar-20	Apr-20	May-20	Jun-20	Jul-20	Aug-20	Sep-20	Oct-19 Sep-20
Proj A	98.9%	130.4%	79.5%	94.8%	98.5%	84.7%	110.9%	93.7%	97.1%	98.1%	75.9%	106.3%	98.8%
Proj B							69.1%	100.0%	69.7%	105.6%	79.9%	81.1%	83.8%
Proj C	100.0%	100.0%	100.0%	100.0%	93.9%	90.2%	90.9%	109.3%	100.0%	98.1%	100.0%	100.0%	98.7%
Proj D	80.8%		66.2%	66.1%		78.3%	98.0%	96.4%	95.7%	82.6%		100.9%	85.0%
Proj E	79.0%	30.8%	152.8%	48.8%	119.2%	35.3%	55.8%	77.4%	98.9%	113.1%	57.9%	97.8%	77.5%
All	**71.7%**	**93.7%**	**79.0%**	**84.6%**	**87.6%**	**80.7%**	**71.0%**	**75.6%**	**83.4%**	**72.4%**	**73.8%**	**74.0%**	**77.6%**

Figure 26: Stoplight Indicator - Estimate Accuracy

At the enterprise level, this summarizes % of planned Stories (or Tasks, or Story Points) Completed by project (rows) for all iterations completing within the calendar reporting month, as well as an enterprise monthly total. Recall that project cadences may vary, so each project monthly stoplight indicator may include 0, 1, 2, or more iterations within that reporting month.

This data can be summarized further into executive-level stoplight dashboards showing the count or percentage of monthly project stoplights, Figure 27, depicting a count of projects meeting stoplight criteria for the current period, previous period, and summary over the past 12 months.

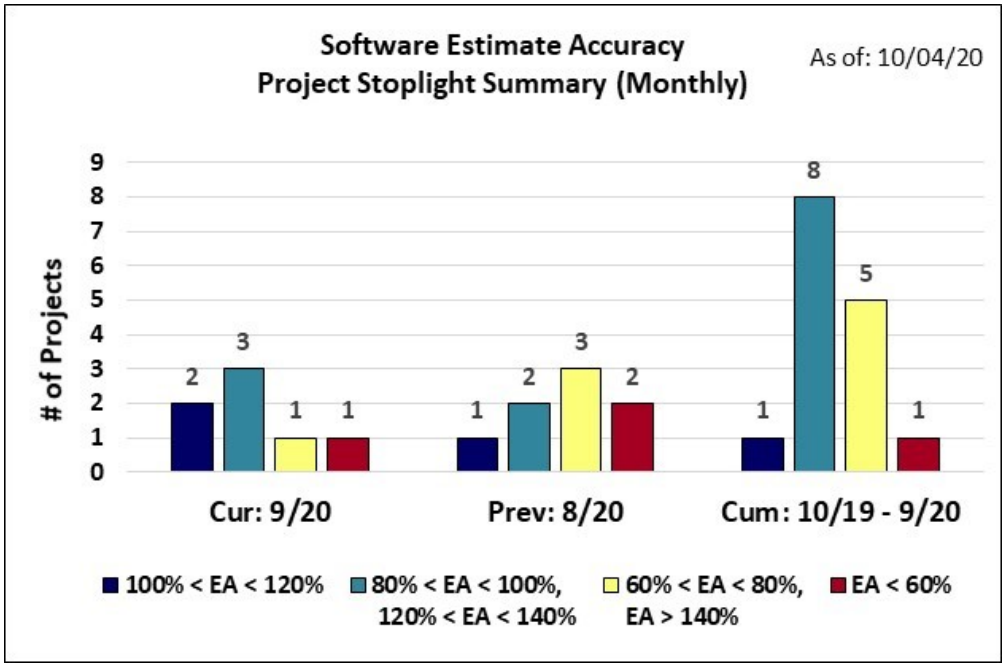


Figure 27: Software Estimate Accuracy Summary

Cumulative Flow (8.4)

Is current capacity keeping up with demand? Is the flow of work proceeding through the value chain?

Cumulative Flow Diagrams (CFDs) are information-rich indicators depicting how efficiently work is being completed across project workflow stages, the rates of task arrivals vs. departures (slopes of CFD bands), and potential queues or backlogs where work rates seem to be slowing. CFDs are generated for project tasks or stories, which are project-specific measures that typically cannot be

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

directly aggregated across projects. At the enterprise level a *Summary* indicator can used to show "mini-CFDs" for a set of projects, based on percentage distribution across workflow states.

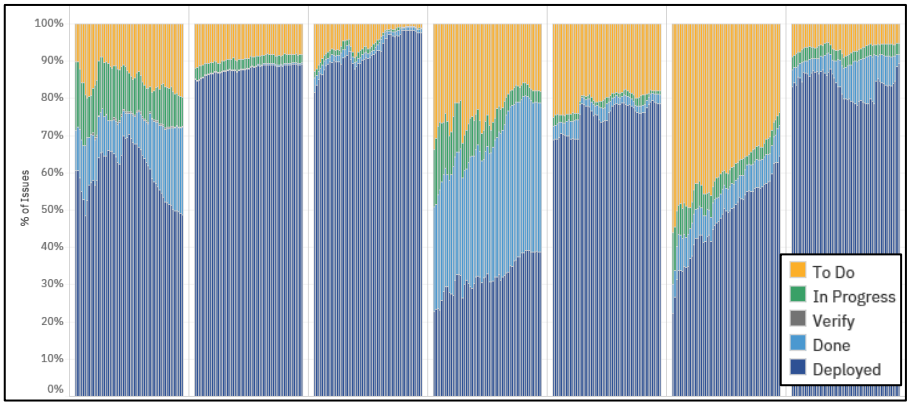


Figure 28: Workflow Indicator

Look for indicators of anomalous workflow distribution, trends, backlogs, ratios of work arrivals vs. departures, and transitions of work to completion states. Distribution will be very dependent on the situation, age and lifecycle state of each project and cannot be compared across projects.

This indicator depends on common workflow states across projects – transformations and mappings of native project workflow states to an enterprise standard such as those shown here may be needed if schemas vary across projects.

Arr/day	Dep/day	Dep/Arr
2.62	1.80	0.69
2.82	2.37	0.84
2.57	2.04	0.79
1.51	1.31	0.87
1.11	1.02	0.92
0.93	1.14	1.22
2.32	1.84	0.79
7.67	6.07	0.79
0.88	0.94	1.06
3.43	3.34	0.97

A healthy demand-driven workflow will have approximately equal rates (parallel CFD bands) of new work (arrival rate) vs. completed work (departure rate), i.e., departures/arrivals ≅1.0. See Figure 15, Notional CFD Diagram for details. Ratios >1.0 indicate the project is deploying completed work at a rate faster than new work is arriving; values <1.0 can indicate a growing backlog of new work received or committed to. Project departure/arrival rates can be depicted in a *Stoplight* dashboard summarizing at a glance how well work completion is balanced vs. the incoming demand of new work, and which projects may need more investigation to ensure efficient workflows and sufficient resources.

Cumulative Flow (CFD) Stoplight X = Average Departures per Day / Average Arrivals per Day			
Red	Yellow	Green	Blue
X < 0.75	0.75 ≤ X < 0.90	0.90 ≤ X ≤1.0	X > 1.0
Implement process efficiencies to increase departures or moderate arrival rate		Monitor or take action as needed	No action

Figure 29: Cumulative Flow Stoplight

Cycle Time / Lead Time (8.5)

How long does it take to implement a feature or capability?

Cycle Time and Lead Time are generally measures and analyses specific to a team, product or environment, measuring the duration from the start of a task (cycle time) or receipt of a customer/user request (lead time) until the task or product is completed and delivered to a baseline or operational environment. (See Figure 2, Measurement Context Diagram) Due to widely varying project or product characteristics it can be difficult to aggregate cycle time measures to the enterprise level. *Summary* indicators can be used to depict cycle times and trends across a set of projects.

These metrics also don't align well with defined-scope type contracts where work does not continuously flow (such as a capacity-based or Time & Materials type contract).

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Defect Detection (8.6)

Defect Resolution (8.7)

How many defects were contained (discovered, saved) prior to internal or external release?

At the project or product level, the intent is to remove defects during development and initial testing of iterations or releases before they escape to impact downstream work.

Defect Resolution Lag Time
As of 19 Dec 19

Defects			(Iteration) 1	2	3	4	5	6	
Defect Discovered (Iteration)	Unknown	0							
	Legacy	0							
	1	82	29	2	19	17	4	11	
	2	123		27	71	6	7	12	
	3	282			122	60	29	71	Blank 0%
	4	112				16	2	94	>1 Iteration 41%
	5	7					5	2	1 Iteration 21%
	6	54						54	Same Iteration 38%
		Total 660	29	29	212	99	47	244	

Measures of overall enterprise defect detection efficiency can be generated in an *Aggregation* indicator very similar to the project-level indicator but summing base measures across projects as an indicator of overall enterprise capability. Trend data provides early warning of performance issues.

In particular, executives often focus on defect escapes, especially those that escape to the field. Executives also want to know if the rate of defects generated is going up over a period of time.

Mean Time to Repair (8.8)

How efficient are we at removing defects once found? How long does it take to restore service?

Enterprise MTTR analysis indicators are similar to those at the project level but reflecting an *Aggregate* of measures across projects. MTTR units (hours, minutes, days) may vary depending on the project mix and operational status. (Figure 30) Analyses is often filtered by defect category to determine closure time for the most critical defects. Root cause analysis can be conducted at the enterprise level to identify opportunities for improvement.

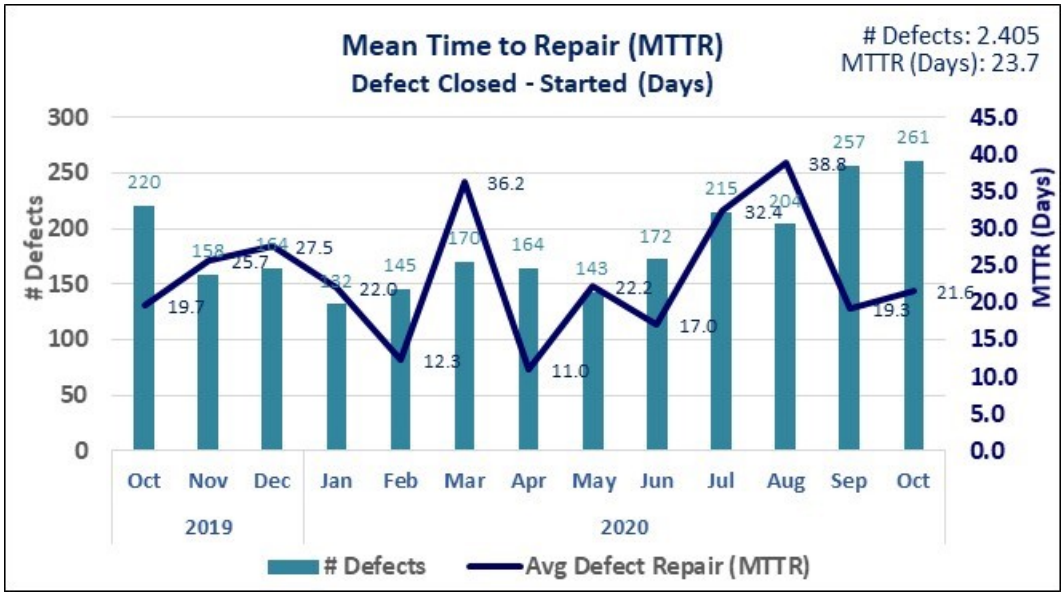


Figure 30: Mean Time to Repair

At executive levels, *Stoplight* indicators (or summary project counts by stoplight color), Figure 31, can be used as flags to managers on how project MTTR measures and trends by reporting period compare to defined enterprise objectives and thresholds.

<= 7 days
<= 15 days
<= 30 days
> 30 days

Different objectives may exist for different Service Level Agreements (SLAs), so emphasis may be on those programs that exceed their agreements.

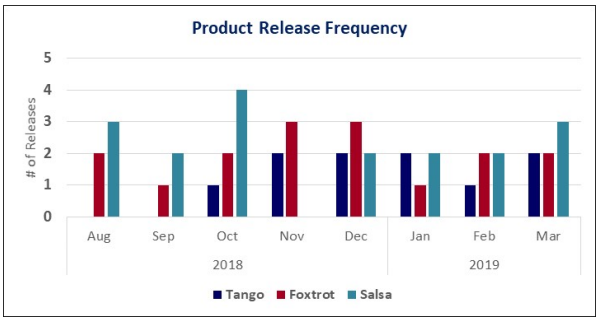
Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

2019	Jan	29.0		2.8	
	Feb			2.4	
	Mar	2.0	3.0	3.7	
<= 7 days	Apr	12.0	15.0	3.9	
<= 15 days	May	10.0		7.3	
<= 30 days	Jun		1.5	10.4	
> 30 days	Jul	2.0	7.5	7.0	
	Aug	18.0	7.7	3.3	4.5
	Sep	37.5	8.1	4.8	5.0
	Oct	18.8	6.0	3.0	8.0
	Nov	3.3	5.5	3.8	5.2
	Dec		1.0	3.0	9.8
2019 Total		20.0	6.1	5.0	6.9
2020	Jan	1.8		5.2	29.7
	Feb	4.5		9.6	2.3
	Mar	6.0	1.0	3.5	2.8
	Apr	4.7	1.7	4.3	29.0
	May	4.0	8.0	2.3	5.4
	Jun		9.0	2.7	12.0
	Jul		2.8	1.0	3.2
	Aug		29.5	6.5	45.4
	Sep	2.0	3.0		22.9
	Oct		2.0		10.0
2020 Total		3.4	11.8	5.7	14.8
Grand Total		15.0	8.4	5.2	13.7

Figure 31: MTTR Stoplight Indicator

Release Frequency (8.9)

How often can we deploy new releases?



The speed and frequency of product releases is often a primary business objective. The Release Frequency indicator spec (8.9) provides an example of an enterprise *Summary* indicator depicting the number of internal or external releases per month by project.

Derived *Aggregate* measures of average release times at the enterprise level can also be generated, but these tend to be more useful for specific teams or products.

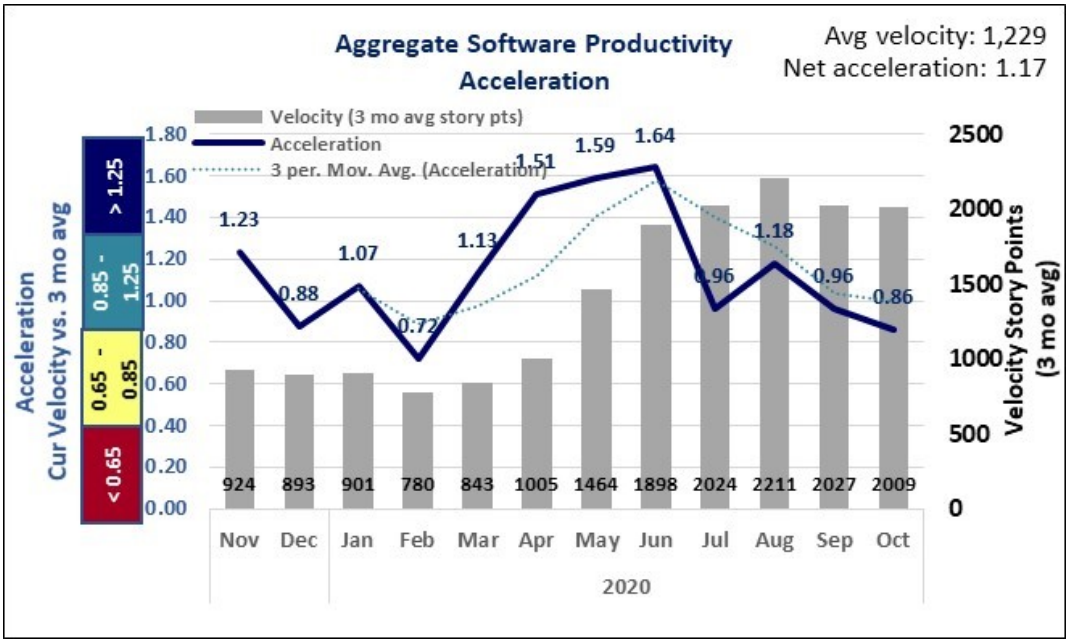
The defined data collection intervals for release reporting may vary based on business need (e.g., annually, quarterly, monthly, weekly, daily). Some enterprises or projects may achieve continuous delivery releases in a near-ops/ ops environment several times per day.

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Velocity (8.10)

Acceleration

Is productivity improving (more work completed per unit time)?



Metric	Measure	Red	Yellow	Green	Blue
Acceleration	Story points completed / (average of last 3 velocities)	<0.65	0.65 - 0.85	0.85 - 1.25	>1.25

Figure 32: Aggregate Software Productivity - Acceleration

Story points and velocity per iteration cannot typically be compared or aggregated across projects due to differing project-specific operational definitions. But as introduced in 8.10, Acceleration is a derived measure that indicates relative changes in Velocity (ratio) compared to an average of recent iterations. Acceleration = 1.0 indicates steady velocity; >1.0 increasing velocity; <1.0 decreasing velocity. Acceleration can be calculated as a project measure, *Aggregate* measure across projects, or as a *Summary* set of "mini-Acceleration" indicators for multiple focus projects. *Stoplight* indicators can also be used at the executive level to summarize project acceleration relative to threshold ranges.

Under stable conditions (staffing, etc.), Acceleration can be used to assess if productivity is improving, declining or steady at the project or enterprise levels. However, since its underlying base measure is story points completed (velocity), Acceleration can be susceptible to variation as the team or product characteristics change. Anomalies are not uncommon and should be understood before taking significant actions.

Table 11: Acceleration

Oct-19	Nov-19	Dec-19	Jan-20	Feb-20	Mar-20	Apr-20	May-20	Jun-20	Jul-20	Aug-20	Sep-20	Oct-20
0.00	1.00	1.64	1.27	1.88	0.58	1.20	0.84	0.95	0.56	0.00	1.00	1.89
1.00	1.17	0.45	1.18	1.17	0.95	1.34	0.91	0.92	1.12	0.54	1.85	0.79
0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.41	0.83	1.05	0.98	0.85	1.13
1.00	0.74	0.83	1.47	1.05	1.00	0.87	1.48	0.69	1.23	0.99	0.72	1.53
1.00	0.23	2.90	2.10	0.31	5.37	0.27	1.09	0.93	0.96	1.78	0.85	1.20
1.00	0.60	2.35	0.45	1.48	0.56	1.37	1.10	0.83	1.57	0.40	1.60	1.34

Use or disclosure of data on this page is subject to the restriction on the copyright page of this report.
Unclassified: Distribution Statement A: Approved for Public Release; Distribution is Unlimited