ESA Software Assurance Workshop

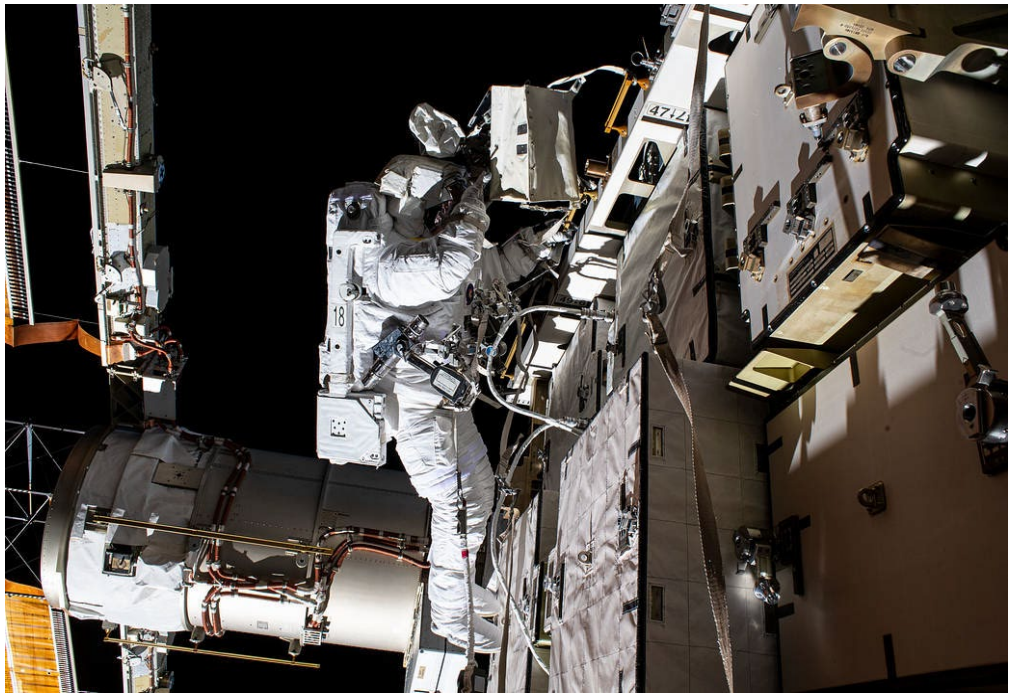
Software Assurance

Tim Crumbley

September 2023

NASA Software Assurance Program Goals

- Provide risk-based performance requirements that provide flexibility for the project Software Assurance and Software Safety activities.
- Improve the risk, issue and finding reporting from the NASA Software Assurance and Software Safety organizations.
- Add value for Software Assurance and Software Safety activities and demonstrate the importance of the NASA Software Assurance activities.
- Numerical Software Quality Assessments (Code and Requirements)
- Provide standard tools and services for Software Assurances activities on projects.
- Provide measurable quantifiable Software Assurance assessments
- Improve the use of data and metrics on all NASA Software Assurance activities.
- Focus Software Assurance activities on known software issues, including targeting Software Assurance and Software Safety research activities.
- Develop more efficient and automated methods for Software Assurance, Software Safety and Software Quality activities.
- Improve Software Assurance training and training requirements in the Safety and Mission Assurance Technical Excellence Program and across the agency.
- Updated documents in software assurance, software safety, and software quality



A Thought!

"What gets measured, gets managed." -

Peter Drucker

- There is so much power in this quote.
- If you've never tracked yourself, you don't even know how much power there is in tracking.
- The simple act of paying attention to something will cause you to make connections you never did before, and **you'll improve those areas - almost without any extra effort.**

Metrics


METRICS
PERFORMANCE
INFORMATION
SEVEN
EFFECTIVENESS
SCORECARD
RELEVANT
MEANINGFUL
UNDERSTANDABLE
RESOURCES
TIME
ANALYSIS
METRIC
SAFETY
VALUE
PRACTICAL
STATISTICAL
COST
MEASUREABLE
SPECIFIC
DATA
MEASURABLE
INDICATOR
ACTIONABLE
STEP
FINANCIAL
HIGH-VALUE
QUALITY
SCOPE
PROCESS
TIME-PHASED
CRITERIA
MEASURES
ACTIONS
STRATEGY
OUTPUTS
CRITICAL
DIRECTIONAL
SERVICE
CONTROL
IMPROVEMENT
ASSESSMENT
LEVELS
ACHIEVABLE


E
1/2
F


0
20
40
60
80
100
120
140
160
180
200
220
240
260
OUTSIDE
TRIP
0.0

Metrics


METRICS
METRIC
PERFORMANCE
INFORMATION
SEVEN
EFFECTIVENESS
SCORECARD
UNDERSTANDABLE
RELEVANT
MEANINGFUL
RESOURCES
TIME
ANALYSIS
SAFETY
VALUE
DATA
PRACTICAL
MEASUREABLE
STATISTICAL
COST
SPECIFIC
SMART
INDICATOR
MEASURABLE
ACTIONABLE
STEP
BALANCED
FINANCIAL
HIGH-VALUE
QUALITY
SCOPE
TIME-PHASED
CRITERIA
PROCESS
MEASURES
ACTIONS
OUTPUTS
CRITICAL
STRATEGY
DIRECTIONAL
LEVELS
ACHIEVABLE
SERVICE
IMPROVEMENT
CONTROL
ASSESSMENT
DEFINITION


E
1/2
F


0
20
40
60
80
100
120
140
160
180
200
220
240
260
OUTSIDE
TRIP
0.0


126.4
lb
TAYLOR

Metrics

TESTS	RESULT	FLAG	UNITS	REFERENCE INTERVAL
Comp. Metabolic Panel (14)				
Glucose	70		mg/dL	65 - 99
BUN	15		mg/dL	6 - 24
Creatinine	1.00		mg/dL	0.76 - 1.27
eGFR If NonAfricn Am	92		mL/min/1.73	>59
eGFR If Africn Am	106		mL/min/1.73	>59
BUN/Creatinine Ratio	15			9 - 20
Sodium	139		mmol/L	134 - 144
Potassium	4.0		mmol/L	3.5 - 5.2
Chloride	100		mmol/L	96 - 106
Carbon Dioxide, Total	25		mmol/L	20 - 29
Calcium	9.0		mg/dL	8.7 - 10.2
Protein, Total	6.5		g/dL	6.0 - 8.5
Albumin	4.0		g/dL	3.5 - 5.5
Globulin, Total	2.5		g/dL	1.5 - 4.5
A/G Ratio	1.6			
Bilirubin, Total	1.0			
Alkaline Phosphatase	45			
AST (SGOT)	40			
ALT (SGPT)	40			

Why Measure?

Management without metrics is just guessing

=

Software Assurance without metrics is just guessing

Candidate Management Indicators Used On Software Projects

- Requirements volatility: total number of requirements and requirement changes over time.
- Bidirectional traceability: Percentage complete of System level requirements to Software Requirements, Software Requirements to Design, Design to Code, Software Requirements to Test Procedures
- Software size: planned and actual number of units, lines of code, or other size measurement over time.
- Software staffing: planned and actual staffing levels over time.
- Software complexity: complexity of each software unit.
- Software progress: planned and actual number of software units designed, implemented, unit tested, and integrated overtime, code developed.
- Problem/change report status: total number, number closed, number opened in the current reporting period, age, severity.
- Software test coverage: a measure used to describe the degree to which the source code of a project is tested by a particular test suite
- Build release content: planned and actual number of software units released in each build.
- Build release volatility: planned and actual number of software requirements implemented in each build.
- Computer hardware and data resource utilization: planned and actual use of computer hardware resources over time.
- Milestone performance: planned and actual dates of key project milestones.
- Scrap/rework: amount of resources expended to replace or revise software products after they are placed under any level of configuration control above the individual author/developer level.
- Effect of reuse: a breakout of each of the indicators above for reused versus new software products.
- Cost performance: identifies how efficiently the project team has turned costs into progress to date.
- Budgeted cost of work performed: identifies the cumulative work that has been delivered to date.
- Audit performance: Are you following a defined processes, how many audits have been completed, audit findings, audit findings open/close numbers
- Risk Mitigation: Number of identified software risks, risk migration status
- Hazard analysis: number of hazard analysis completed, hazards mitigation steps addressed in software requirements and design, number of mitigation steps tested

Objectives of the Software Assurance and Software Safety Standard

a. Ensuring that the processes, procedures, and products used to produce and sustain the software conform to all specified requirements and standards that govern those processes, procedures, and products.

   (1) A set of activities that assess adherence to, and the adequacy of the software processes used to develop and modify software products.

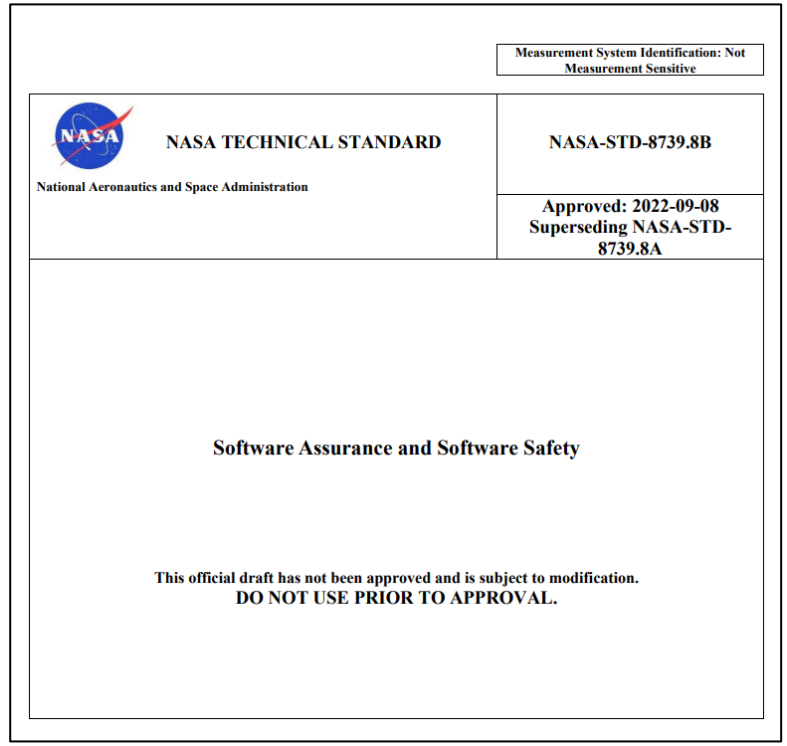
   (2) A set of activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the software processes are appropriate for and produce software products of suitable quality for their intended purposes.

b. Determining the degree of software quality obtained by the software products.

c. Ensuring that the software systems are safe and that the software safety-critical requirements are followed.

d. Ensuring that the software systems are secure.

e. Employing rigorous analysis and testing methodologies to identify objective evidence and conclusions to provide an independent assessment of critical products and processes throughout the life cycle.



Measurement System Identification: Not Measurement Sensitive

NASA TECHNICAL STANDARD

National Aeronautics and Space Administration

NASA-STD-8739.8B

Approved: 2022-09-08
Superseding NASA-STD-8739.8A

Software Assurance and Software Safety

This official draft has not been approved and is subject to modification.
DO NOT USE PRIOR TO APPROVAL.

https://swehb.nasa.gov/

Key Software Assurance Focus Areas

- Planning
  - Implementation of the NASA-STD-8739.8A requirements
  - Software assurance\safety requirements mapping matrix, review any tailored requirements
  - Software assurance\safety approach, plan and resource allocations
  - "Software assurance\safety requirements flow down into contracts"

- "Analysis"
  - Software requirements analysis
  - Software safety analysis
  - Software test analysis
  - Software hazard analysis
  - Software source code quality analysis
  - Peer reviews

- Static Analysis Tools Assessments

- Audits
  - Software engineering requirements flow down and implementation
  - Software process audits
  - Software test witnessing

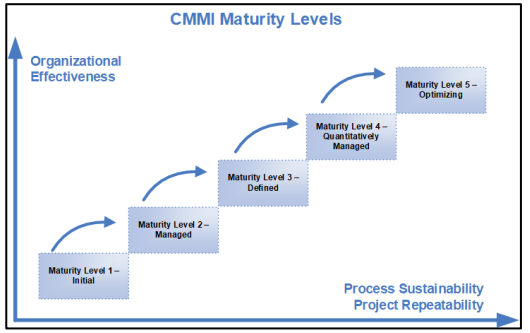
- Communication
  - Software assurance and software safety planned activities
  - Metric and status reporting by software assurance\safety
  - IV&V plan and communication (if required)
  - Software risks, findings or known issues

- Product reviews
  - Major Milestone product reviews
  - Software development product reviews
  - Software metric data reviews

- Defect Tracking and Management
  - Root causes analysis

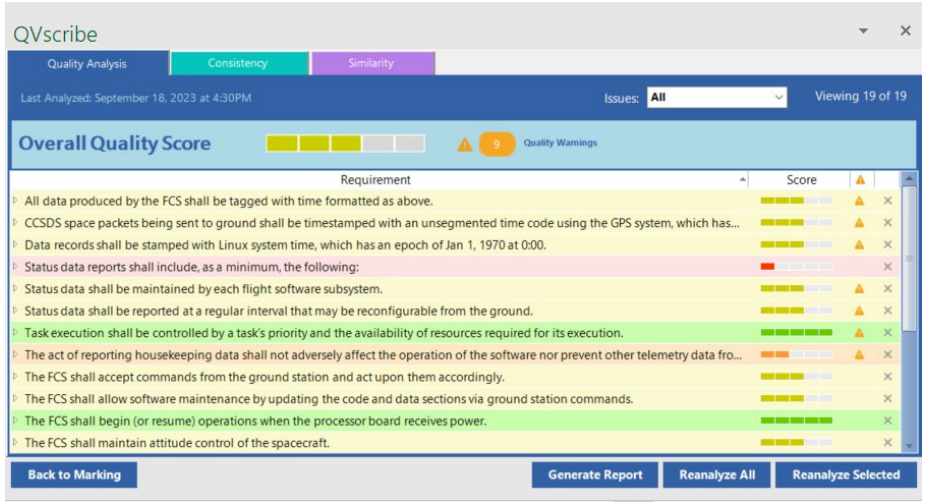
Key Software Assurance Focus Areas

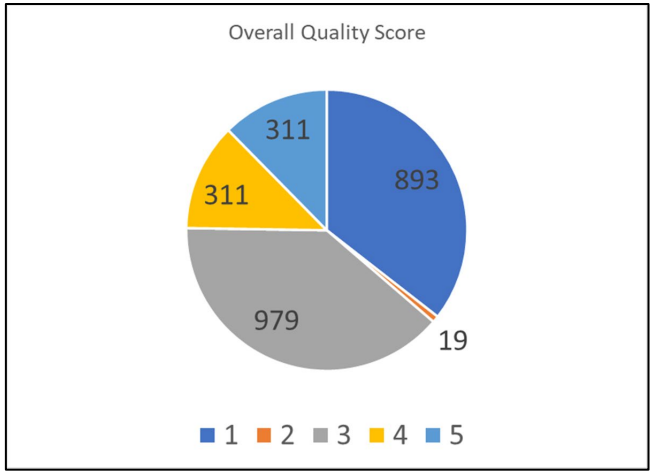
- **Planning (Ensuring Processes and Determining Quality)**
  - Implementation of the Software Assurance and Software Safety Standard, NASA-STD-8739.8, requirements
  - Software assurance\safety requirements mapping matrix, review any tailored requirements
    - Tailoring of the specific requirements in NASA-STD-8739.8
      - # of projects tailoring each requirement
      - % of requirements tailored per project
  - Software assurance\safety approach, plan and resource allocations
    - Trending of Software Assurance cost estimates throughout life-cycle
    - Software assurance resource utilization throughout life-cycle
  - Software assurance\safety requirements flow down into contracts
    - Process Maturity of the software development organization



Key Software Assurance Focus Areas

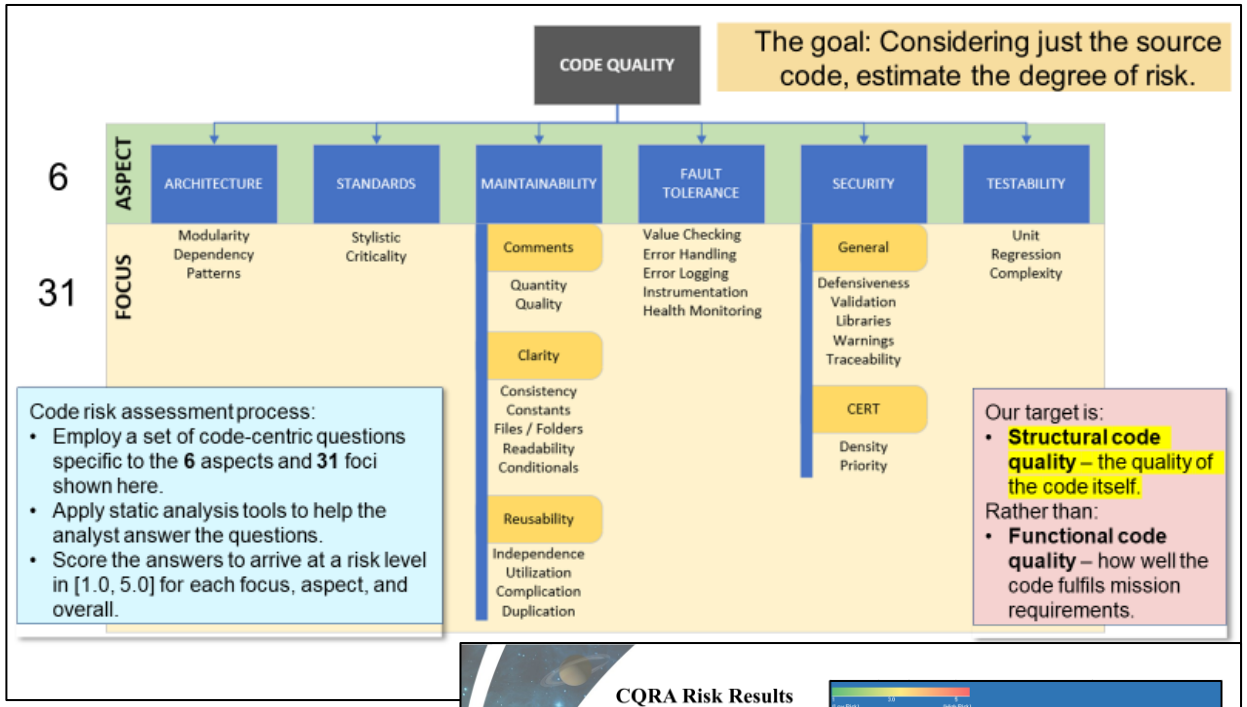
- Analysis (Processes, Quality, Safe, and Analysis)
  - Software requirements analysis
    - Software requirements quality risk score
    - Software Requirements Volatility trend
    - # of TBD/TBC/TBR in the software requirements
    - # requirements vs number of developed lines of code
  - Software safety and hazard analysis
    - Percentage of the software hazards that have defined completed causes and verification approaches
    - # of software requirements tracing to software hazards

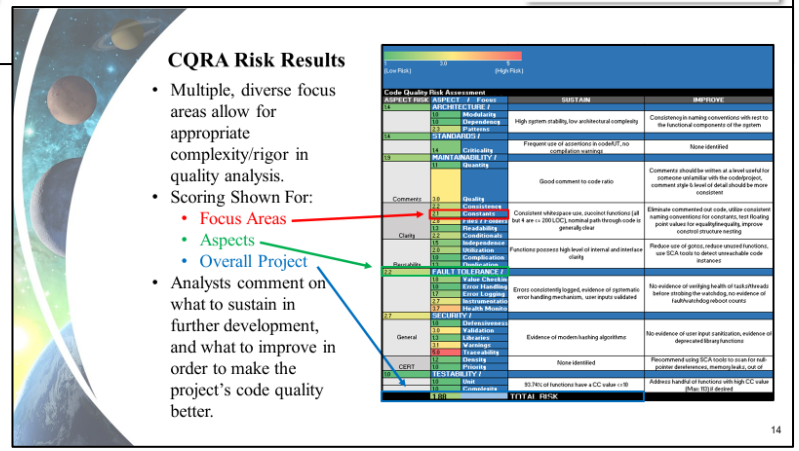




Key Software Assurance Focus Areas

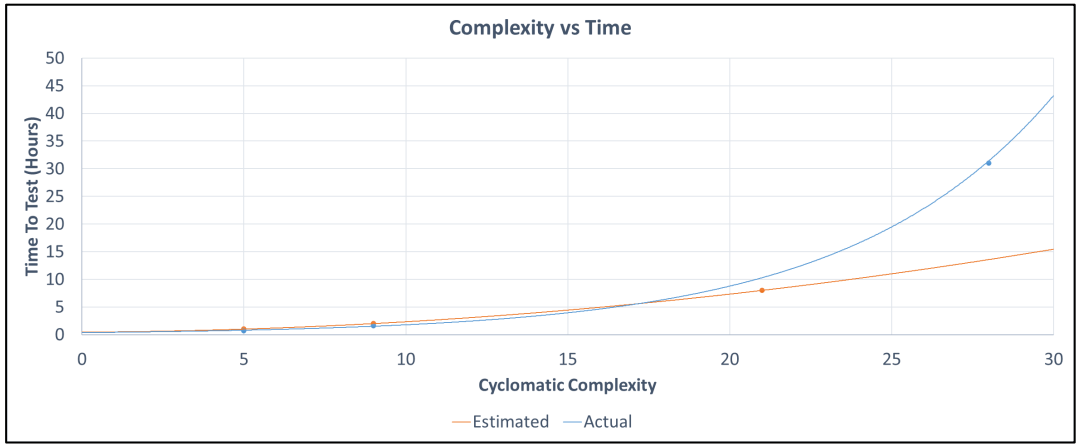
- Analysis
  - Software design analysis
    - (Ensuring processes and Determining quality)
      - Software data dictionary fields are correct. % of the Data dictionary data definitions that are complete.
      - % of design functions that traces to the software requirements
  - Software source code quality analysis
    - (Determining quality)
      - Code Quality Risk Assessment Scores
      - Software cyclomatic complexity

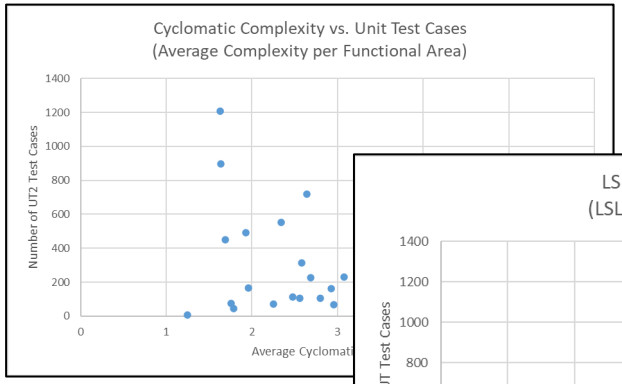


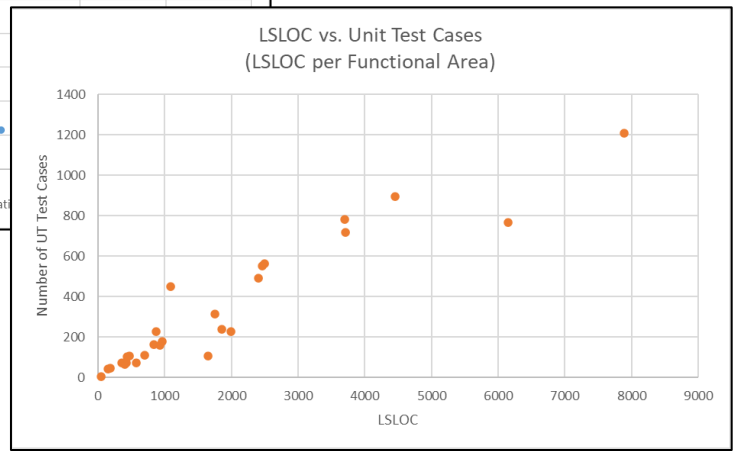


Key Software Assurance Focus Areas

- Analysis
  - Software test analysis (Processes, Quality, Secure, and Analysis)
    - Code coverage data: % of code that has been executed during testing
    - % of software requirements that have been verified or tested
    - % of software test results reviewed by software assurance
    - # of independent software tests run by software assurance and IV&
  - Peer reviews (Processes, Quality)
    - # of Peer Review Audits planned vs. # of Peer Review Audits performed
    - # of Non-Conformances identified in each peer review

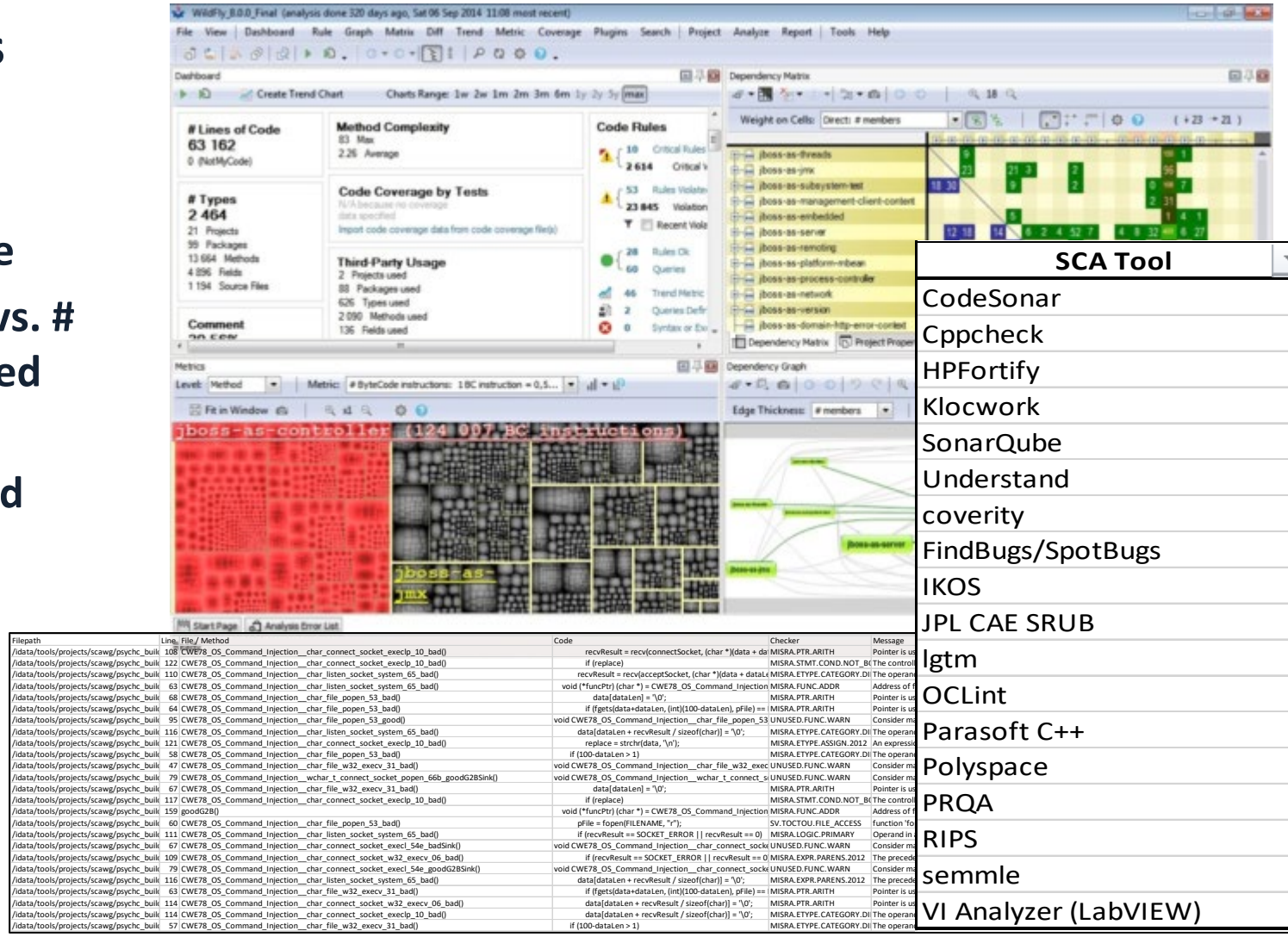






Key Software Assurance Focus Areas

- **Static Analysis Tools Assessments (Quality, Secure, and Analysis)**
  - Software cyclomatic complexity
  - # of static analysis tools used to date
  - # of errors and warnings evaluated vs. # of total errors and warnings identified by each tool
  - # of Cybersecurity vulnerabilities and weaknesses
  - Total # of static code analysis "positives" vs. # of "positives" resolved. Trend over time.



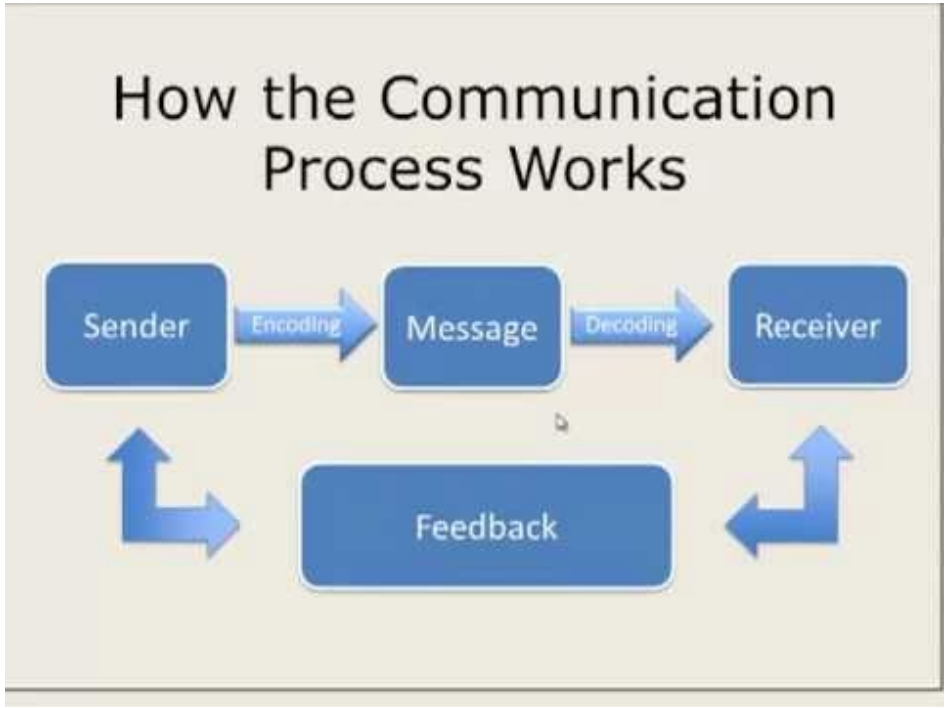
SCA Tool
CodeSonar
Cppcheck
HPFortify
Klocwork
SonarQube
Understand
coverity
FindBugs/SpotBugs
IKOS
JPL CAE SRUB
lgtm
OCLint
Parasoft C++
Polyspace
PRQA
RIPS
semmle
VI Analyzer (LabVIEW)

Key Software Assurance Focus Areas

- Audits (Processes, Quality, and Analysis)
  - Software process audits
    - # of Audits conducted by the project – Planned vs. Actual.
    - # of software work product Non-Conformances identified by audits
    - % of the software processes that have been audited
  - Software test witnessing
    - % of software test witnessed
    - # of findings identied in software test witnessing

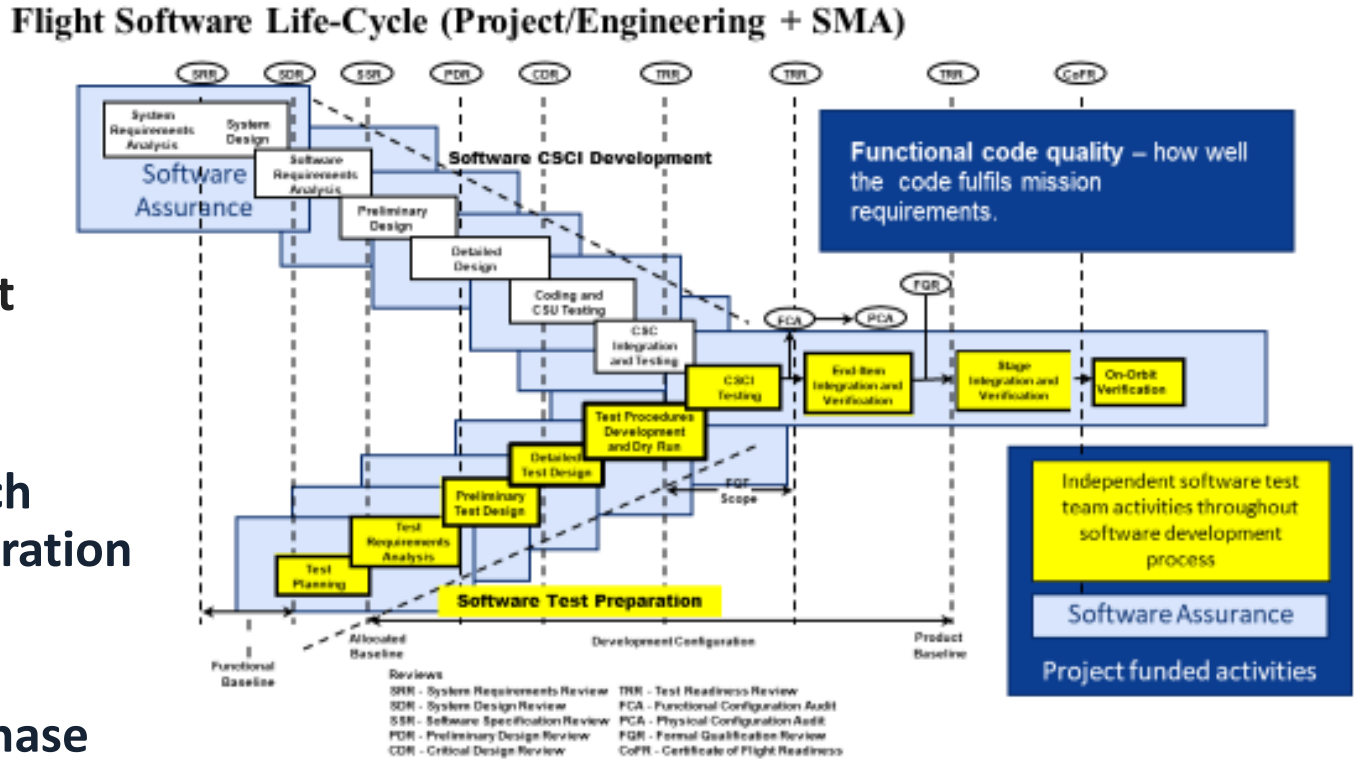
Key Software Assurance Focus Areas

- **Communication (Processes, Quality, and Analysis)**
  - Software assurance and software safety planned activities
    - Completed\in progress\not started software assurance and software safety activities
  - Metric and status reporting by software assurance\safety
    - # of Defect or Problem Reports identified by SA vs. total # Defect or Problem Reports
  - IV&V plan and communication
    - # of Severity 1 or 2 IV&V findings
      - Found
      - Addressed by the project
  - Software risks, findings or known issues
    - Total # of Non-Conformances over time (Open, Closed, # of days Open, and Severity of Open)



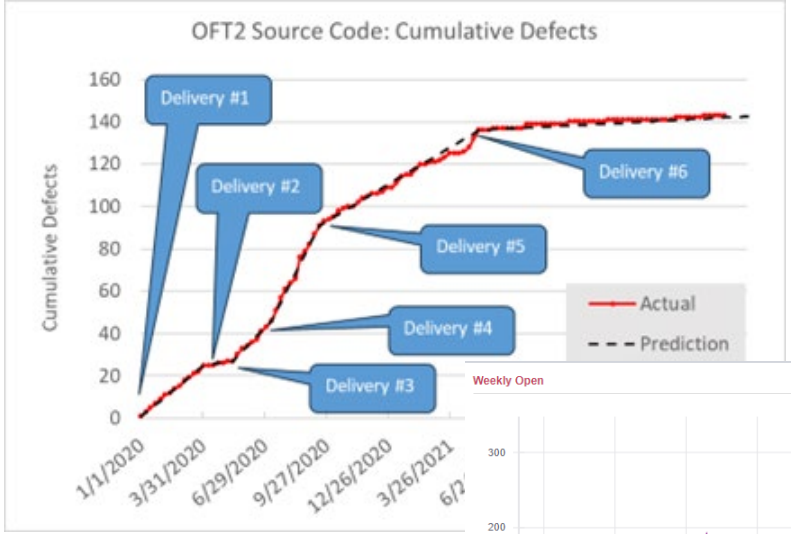
Key Software Assurance Focus Areas

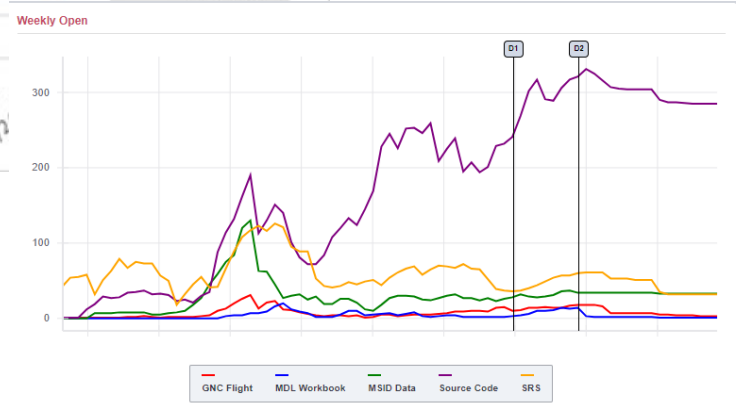
- **Product reviews (Processes, Quality, and Analysis)**
  - Major Milestone product reviews
    - # of RFAs/RIDs identied by Software Assurance
    - # of RFAs/RIDs identified by milestone
    - Status of the software entrance and exit criteria meet at each review point
  - Software development product reviews
    - # of software Non-Conformances at each Severity level for each software configuration item (Open, Closed)
    - # of software work product Non-Conformances identified by life-cycle phase over time



Key Software Assurance Focus Areas

- **Defect Tracking and Management (Processes, Quality, and Analysis)**
  - # of issues open
  - # of Cybersecurity vulnerabilities and weaknesses identified
  - # of open defects per release by severity level
- **Root causes analysis**
  - # of software work product Non-Conformances identified by life-cycle phase over time
  - # of software defects per cause item





Summary

Components	Arrival		Open	
	Defects	% Defect	Defects	% Open
GNC Flight	112	4.3%	3	2.7%
MDL Workbook	101	3.8%	1	1%
MSID Data	327	12.4%	33	10.1%
Source Code	1528	58%	285	18.7%
SRS	567	21.5%	32	5.6%
Total	2635	100%	354	13.4%

Repeat The Thought

"What gets measured, gets managed." - Peter Drucker

The simple act of paying attention to something will cause you to make connections you never did before, and **you'll improve those areas** - almost without any extra effort.

Repeat The Thought

"What gets measured, gets managed." - Peter Drucker

The simple act of paying attention to something will cause you to make connections you never did before, and **you'll improve those areas** - almost without any extra effort.

So, what are your currently measuring or paying attention to on your software assurance project?

Questions?

Tim Crumbley - NASA Software Assurance Tech Fellow
Cell: 256.783.5912
Email: tim.crumbley@nasa.gov