Test Metrics:
A Practical Approach to Tracking & Interpretation

Presented By:
Shaun Bradshaw
Questcon Technologies

sbradshaw@questcon.com

QUESTCON Technologies

Objectives

- Why Measure?
- Definition
- Metrics Philosophy
- Types of Metrics
- Interpreting the Results
- Metrics Case Study
- Q & A

Why Measure?

"Software bugs cost the U.S. economy
an estimated $59.5 billion per year.
An estimated $22.2 billion
could be eliminated by
improved testing that enables
earlier and more effective
identification and removal of defects."

- US Department of Commerce (NIST)

Why Measure?

It is often said,

"You cannot improve what you cannot measure."

Definition

Test Metrics:

- Are a standard of measurement.
- Gauge the effectiveness and efficiency of several software development activities.
- Are gathered and interpreted throughout the test effort.
- Provide an objective measurement of the success of a software project.

Metrics Philosophy

When tracked and used properly, test metrics can aid in software development process improvement by providing pragmatic & objective evidence of process change initiatives.

Metrics Philosophy

Keep It Simple

Make It Meaningful

Track It

Use It

- Measure the basics first
- Clearly define each metric
- Get the most “bang for your buck”

Metrics Philosophy

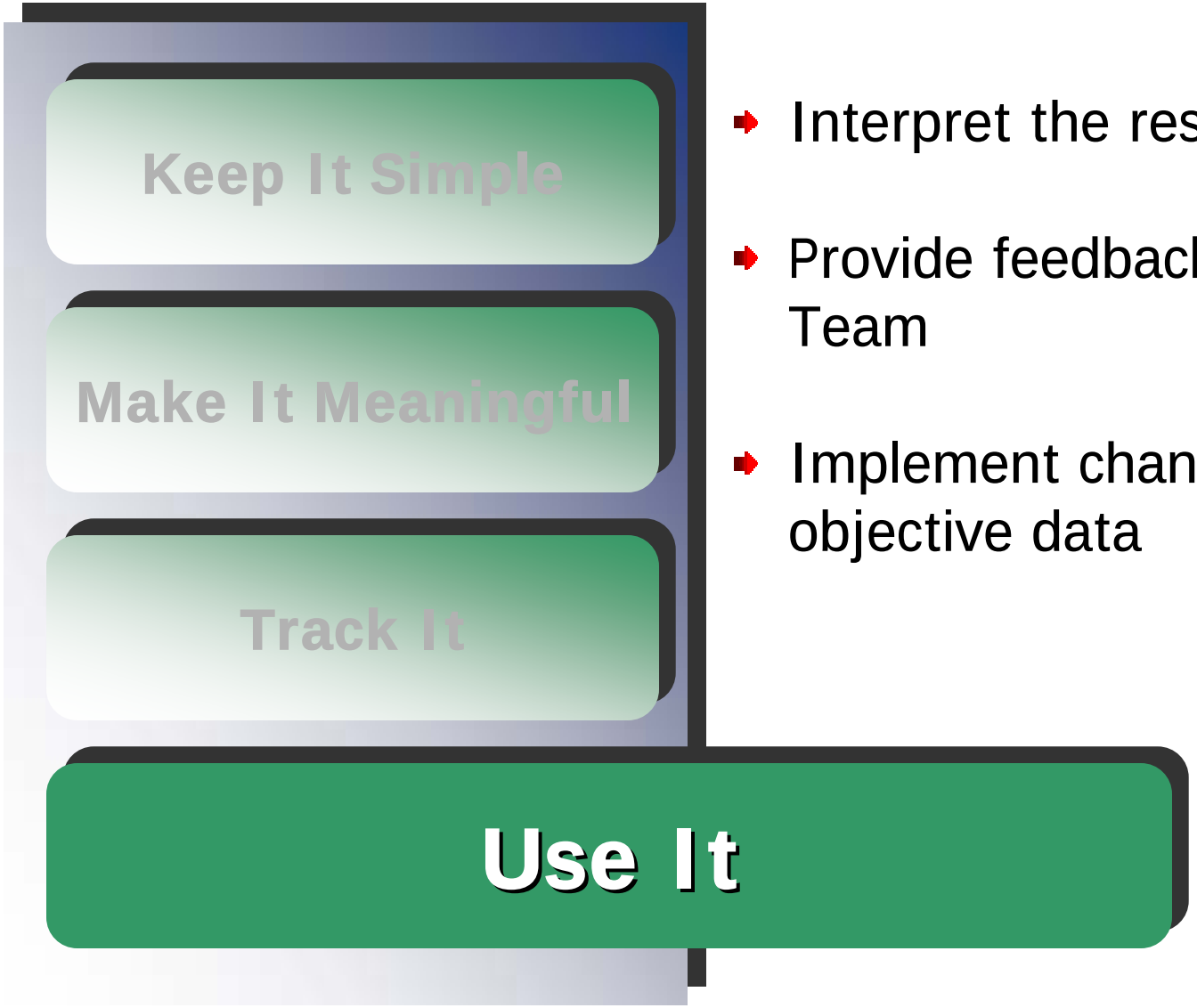
- Metrics are useless if they are meaningless (use GQM model)
- Must be able to interpret the results
- Metrics interpretation should be objective

Metrics Philosophy

- Incorporate metrics tracking into the Run Log or defect tracking system
- Automate tracking process to remove time burdens
- Accumulate throughout the test effort & across multiple projects

QUESTCON Technologies

Metrics Philosophy



- Interpret the results
- Provide feedback to the Project Team
- Implement changes based on objective data

Types of Metrics

Base Metrics

- Raw data gathered by Test Analysts
- Tracked throughout test effort
- Used to provide project status and evaluations/feedback

Examples

- # Test Cases
- # Executed
- # Passed
- # Failed
- # Under Investigation
- # Blocked
- # 1st Run Failures
- # Re-Executed
- Total Executions
- Total Passes
- Total Failures

Types of Metrics

Base Metrics

- Raw data gathered by Test Analyst
- Tracked throughout test effort
- Used to provide project status and evaluations/feedback

Blocked

- The number of distinct test cases that cannot be executed during the test effort due to an application or environmental constraint.
- Defines the impact of known system defects on the ability to execute specific test cases

Examples

- # Test Cases
- # Executed
- # Passed
- # Failed
- # Under Investigation
- **# Blocked**
- # 1st Run Failures
- # Re-Executed
- Total Executions
- Total Passes
- Total Failures

Types of Metrics

Calculated Metrics

- Tracked by Test Lead/Manager
- Converts base metrics to useful data
- Combinations of metrics can be used to evaluate process changes

Examples

- % Complete
- % Test Coverage
- % Test Cases Passed
- % Test Cases Blocked
- 1st Run Fail Rate
- Overall Fail Rate
- % Defects Corrected
- % Rework
- % Test Effectiveness
- Defect Discovery Rate

Types of Metrics

Calculated Metrics

- Tracked by Test Lead/Manager
- Converts base metrics to useful data
- Combinations of metrics can be used to evaluate process changes

1st Run Fail Rate

- The percentage of executed test cases that failed on their first execution.
- Used to determine the effectiveness of the analysis and development process. Comparing this metric across projects shows how process changes have impacted the quality of the product at the end of the development phase.

Examples

- % Complete
- % Test Coverage
- % Test Cases Passed
- % Test Cases Blocked
- **1st Run Fail Rate**
- Overall Fail Rate
- % Defects Corrected
- % Rework
- % Test Effectiveness
- Defect Discovery Rate

Sample Run Log

Sample System Test

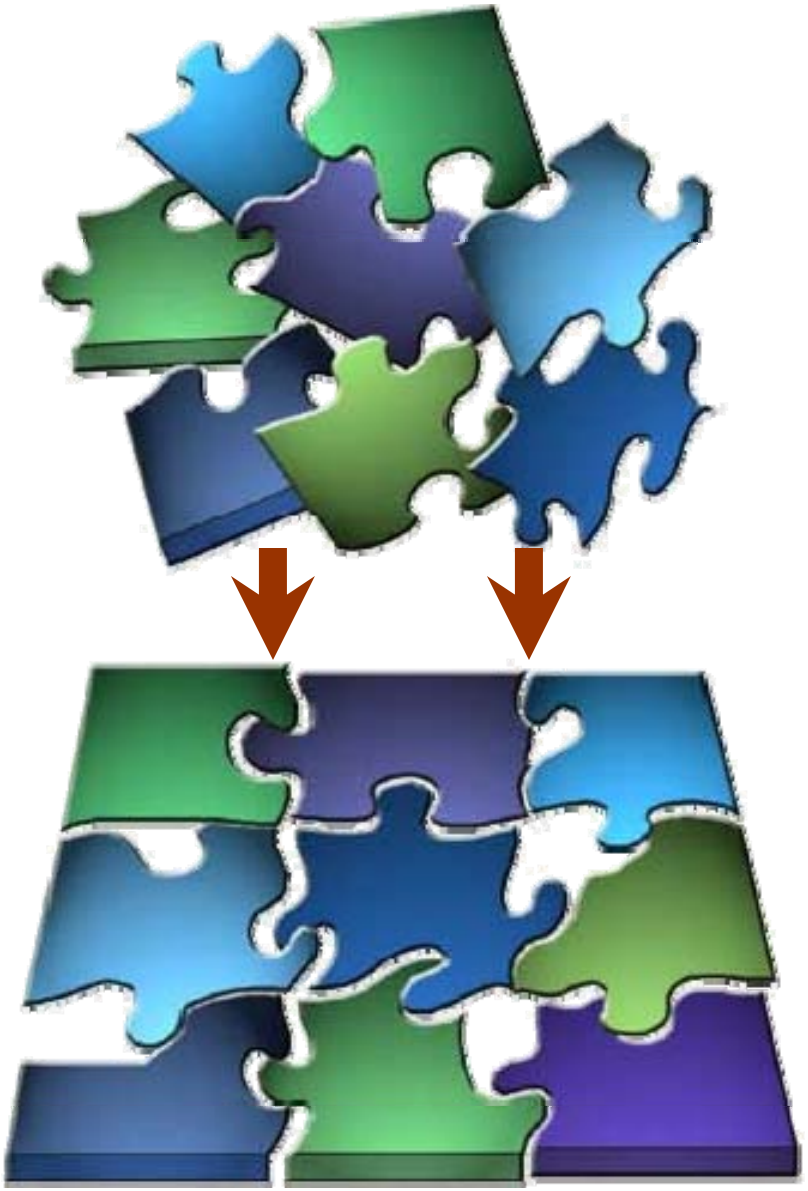
TC ID	Run Date	Actual Results	Run Status							Current Status	# of Runs
SST-001	01/01/04	Actual results met expected results.	P							P	1
SST-002	01/01/04	Sample failure	F							F	1
SST-003	01/02/04	Sample multiple failures	F	F	P					P	3
SST-004	01/02/04	Actual results met expected results.	P							P	1
SST-005	01/02/04	Actual results met expected results.	P							P	1
SST-006	01/03/04	Sample Under Investigation	U							U	1
SST-007	01/03/04	Actual results met expected results.	P							P	1
SST-008		Sample Blocked	B							B	0
SST-009		Sample Blocked	B							B	0
SST-010	01/03/04	Actual results met expected results.	P							P	1
SST-011	01/03/04	Actual results met expected results.	P							P	1
SST-012	01/03/04	Actual results met expected results.	P							P	1
SST-013	01/03/04	Actual results met expected results.	P							P	1
SST-014	01/03/04	Actual results met expected results.	P							P	1
SST-015	01/03/04	Actual results met expected results.	P							P	1
SST-016											0
SST-017											0
SST-018											0
SST-019											0
SST-020											0

Sample Run Log

Base Metrics	
Metric	**Value**
Total # of TCs	100
# Executed	13
# Passed	11
# Failed	1
# UI	1
# Blocked	2
# Unexecuted	87
# Re-executed	1
Total Executions	15
Total Passes	11
Total Failures	3
1st Run Failures	2

Calculated Metrics	
Metric	**Value**
% Complete	11.0%
% Test Coverage	13.0%
% TCs Passed	84.6%
% TCs Blocked	2.0%
% 1st Run Failures	15.4%
% Failures	20.0%
% Defects Corrected	66.7%
% Rework	100.0%

Metrics Program – No Analysis



Issue:

The test team tracks and reports various test metrics, but there is no effort to analyze the data.

Result:

Potential improvements are not implemented leaving process gaps throughout the SDLC. This reduces the effectiveness of the project team and the quality of the applications.

Metrics Analysis & Interpretation

Solution:

- Closely examine all available data
- Use the objective information to determine the root cause
- Compare to other projects
  - ✓ Are the current metrics typical of software projects in your organization?
  - ✓ What effect do changes have on the software development process?

Result:

Future projects benefit from a more effective and efficient application development process.

QUESTCON Technologies

Metrics Case Study

VOLVO

Volvo IT of North America had little or no testing involvement in its IT projects. The organization's projects were primarily maintenance related and operated in a COBOL/CICS/Mainframe environment. The organization had a desire to migrate to newer technologies and felt that testing involvement would assure and enhance this technological shift.

While establishing a test team we also instituted a metrics program to track the benefits of having a QA group.

Metrics Case Study

VOLVO

Project V

- Introduced a test methodology and metrics program
- Project was 75% complete (development was nearly finished)
- Test team developed 355 test scenarios
- 30.7% - 1st Run Fail Rate
- 31.4% - Overall Fail Rate
- Defect Repair Costs = $519,000

Metrics Case Study

VOLVO

Project T

- Instituted requirements walkthroughs and design reviews with test team input
- Same resources comprised both project teams
- Test team developed 345 test scenarios
- 17.9% - 1st Run Fail Rate
- 18.0% - Overall Fail Rate
- Defect Repair Costs = $346,000

Metrics Case Study

VOLVO

	Project V	Project T	Reduction of
1st Run Fail Rate	30.7%	17.9%	41.7%
Overall Failure Rate	31.4%	18.0%	42.7%
Cost of Defects	$ 519,000.00	$ 346,000.00	$ 173,000.00

Reduction of **33.3%** in the cost of defect repairs

Every project moving forward, using the same QA principles can achieve the same type of savings.

QUESTCON Technologies

Q & A

QUESTCON Technologies